



## TABLE OF CONTENT

ACKNOWLEDGEMENT .....	3
ABSTRACT .....	4
LIST OF FIGURES .....	5
LIST OF TABLES .....	5
CHAPTER 1 INTRODUCTION .....	6
1.1 What is a whiteboard? .....	6
1.2 Why is it needed? .....	7
1.3 How is it used? .....	10
1.4 Why is it important? .....	12
1.5 What are the benefits? .....	16
1.6 Summary .....	17
CHAPTER 2 LITERATURE REVIEW .....	18
2.1 Introduction .....	18
2.2 Whiteboard .....	19
2.3 Whiteboard .....	22
2.3.1 Whiteboard .....	22
2.3.2 Whiteboard .....	22
2.3.3 Whiteboard .....	28
2.3.4 Whiteboard .....	35
2.4 Whiteboard .....	37
2.4.1 Whiteboard .....	37
2.4.2 Whiteboard .....	39
2.4.3 Whiteboard .....	41
2.4.4 Whiteboard .....	42
2.4.5 Whiteboard .....	42
2.4.6 Whiteboard .....	42
2.4.7 Whiteboard .....	45
2.4.8 Whiteboard .....	48
2.4.9 Whiteboard .....	48
2.4.10 Whiteboard .....	49
2.4.11 Whiteboard .....	50
2.4.12 Whiteboard .....	52
2.4.13 Whiteboard .....	52
2.4.14 Whiteboard .....	53
2.4.15 Whiteboard .....	54
2.4.16 Whiteboard .....	55
2.4.17 Whiteboard .....	55



## TABLE OF CONTENT

ACKNOWLEDGEMENT .....	3
ABSTRACT .....	4
LIST OF FIGURES .....	5
LIST OF TABLES .....	5
CHAPTER 1: INTRODUCTION .....	6
1.1 What is whiteboard? .....	6
1.2 Problem Definition.....	7
1.3 Project Objective .....	10
1.4 Project Scope .....	12
1.5 Project Limitation .....	16
1.6 Summary .....	17
CHAPTER 2: LITERATURE REVIEW .....	18
2.1 Purpose .....	18
2.2 Information gather and source .....	19
2.3 Development Models .....	22
2.3.1 System Engineering.....	22
2.3.2 Waterfall Model .....	22
2.3.3 Prototype Model .....	28
2.3.4 The Spiral Model .....	35
2.4 Software Developments and Tools .....	37
2.4.1 Visual Basic .....	37
2.4.2 Java .....	39
2.5 Summary.....	41
CHAPTER 3: METHODOLOGY .....	42
3.1 System Analysis .....	42
3.1.1 Prototyping Development Method .....	42
3.1.2 Reasons I Used Prototyping Model .....	45
3.2 System Requirements.....	48
3.2.1 Functional Requirements .....	48
3.2.2 Non-Functional Requirements .....	49
3.2.3 Hardware and Software Requirement .....	50
3.3 System Design .....	52
3.3.1 System Architecture Design .....	52
3.3.2 Whiteboard module Architecture Design .....	53
3.3.3 System Modeling.....	54
3.3.4 System Functionality Design .....	55
3.3.5 Whiteboard Module Functionality Design .....	58





3.4	User Interface Design .....	60
3.5	Project Scheduling .....	66
3.6	Statements of Expected Outcome .....	66
CHAPTER 4: System Implementation .....		68
4.1	Standards and procedures .....	68
4.2	System Design .....	70
4.3	Interface .....	73
4.4	Sample Coding .....	78
CHAPTER 5: TESTING .....		84
5.1	Testing Fundamental .....	84
5.2	Testing Technique .....	86
5.2.1	Unit Testing .....	86
5.2.3	Bottom-Up integration .....	87
5.2.3	Test Case .....	89
5.3	Debugging Programs .....	91
5.3.1	Types of bugs .....	91
5.3.2	The VB debugger .....	91
PROJECTS DISCUSSION .....		92
CONCLUSION .....		93
REFERENCES .....		95
APPENDIX .....		96
Getting Start On White Board .....		96
The Whiteboard .....		96
The Tools Bar .....		98
The Color Control .....		100
The Graphic Effects .....		101
The Pen Size Control .....		104
Status Bar .....		104
Saving and Printing .....		105



## LIST OF FIGURES

	<b>Page</b>
<i>Figure 2.1: The Waterfall Model of software development</i>	28
<i>Figure 2.2: The Prototype model of software development</i>	35
<i>Figure 2.3: The Spiral Model of software development</i>	36
<i>Figure 3.1: Prototyping Model Requirement Analysis</i>	38
<i>Figure 3.2: Prototype Model</i>	46
<i>Figure 3.3: system architecture design</i>	47
<i>Figure 3.4: The whiteboard module architecture design</i>	53
<i>Figure 3.5 Server Mode Context Level Diagram</i>	55
<i>Figure 3.6 Server Mode Level 0 DFD</i>	57
<i>Figure 3.7: Communication tools process</i>	58
<i>Figure 3.8: Login form</i>	60
<i>Figure 3.9: Session Layout</i>	61
<i>Figure 3.10: Chat Layout</i>	62
<i>Figure 3.11: Paint Layout</i>	63
<i>Figure 3.12: Audio Layout</i>	64
<i>Figure 3.13 User Preference Layout</i>	65
<i>Figure 4.1: Finalize White Board Module DFD</i>	70
<i>Figure 4.2: DFD for sub module Tools</i>	71
<i>Figure 4.3: Initial interface prototyping</i>	73
<i>Figure 4.4: Second implementation of design interface</i>	74
<i>Figure 4.5: Finalize User Interface</i>	74
<i>Figure 5.1: Levels of testing</i>	86
<i>Figure 5.2: bottom up integration of White Board module</i>	88

## LIST OF TABLES

<i>Table 3.1: Hardware and software requirements</i>	61
<i>Table 3.2: Project Schedule</i>	66
<i>Table 4.1: Simplified Class Diagram</i>	69
<i>Table 4.1: Simplified Class Diagram</i>	78
<i>Table 5.1: Test Case for White Board module</i>	89
<i>Table 5.2: Method of controlling program execution</i>	92





## ACKNOWLEDGEMENT

The Network Integrated Whiteboard System is a audio integrated, graphics and network related application. This application creates a new pace on real-time presentation integrated with voice conferencing through networks, which are suitable for lecture presentation and Computer Based Education (CBE). This huge system will divide into three subsystems or modules, and our team, which are constructed with three members, assign each subsystem to every member. The three subsystem mentioned are the whiteboard module, voice communication module and the server module. Mr. Wong Chan Weng who is in charge in the voice communication module, while Mr. Lee Chee Ching will cover the server module and I myself will be in the whiteboard module.

Besides, I would like to express my appreciation and thankful to all the great folks and course mates for their terrific support and help – especially Mr. Lee Chee Ching and Mr. Wong Chang Weng who are my team partners, Mr. Ng Yih Young and Mr. Lim Boon Hong who are also my best friends helping me in information gather and system analysis. Finally, the completion of this report depends largely on the resources available through the Internet, reference books, and past year thesis reports.



## **ABSTRACT : INTRODUCTION**

The Network Integrated Whiteboard System is a audio integrated, graphics and network related application. This application creates a new pace on real-time presentation integrated with voice conferencing through networks, which are suitable for lecture presentation and Computer Based Education (CBE).

This huge system will divide into three subsystems, or modules, and our team, which are constructed with three members, assign each subsystem to every member. The three subsystem mentioned are the whiteboard module, voice communication module and the server module. Mr. Wong Chan Weng will in charge in the voice communication module while Mr. Lee Chee Ching will cover the server module and I myself will involve in the whiteboard module.

The whiteboard module consists of many sub-modules like drawing, writing and typing, imaging, pointing, and graphic effects. My tasks also involve design of graphic user interface (GUI) of the system and make integration of the other modules. We do lots of researches on printed resources or non-printed resources mainly to develop a reliable, secure, maintainable, modifiable, efficient whiteboard application.





## CHAPTER 1: INTRODUCTION

### 1.1 What is whiteboard?

The Concise Oxford Dictionary defines whiteboard as a board applications (electronic whiteboard, or just e - board) and the second with a white surface, used especially for classroom presentation using felt-tip pens. A whiteboard is simply a piece board with a surface on which an erasable colored marking pen can be used to write a brief outline of notes and to sketch graphs results. Normally, peoples who lead a presentation like lecturers, teachers, sales and marketing people (simply call the presenter) are the users of whiteboard.

New technologies at the current day implements whiteboard to overcome these requirements. It is only an application operated in a become electronic like projector, with some off - the - shelf applications like Microsoft PowerPoint and some Paint-related applications. There have another implementation, which technologies internetwork being used nowadays, as we can see, through the Internet or Intranet owned by a corporation for business or research purpose. By the way, these create a wide exposures in create a high - performance, reliable, maintainable, secure application.



## 1.2 Problem Definition

We define the problem into two separate sections: first, comparison between the real whiteboard and the off – the – shelf whiteboard on this net - board at the same time, or one is drawing and the other applications (electronic whiteboard, or just e - board) and the second one (said intruder) erasing it at the same time, how can this is, the problem within this applications.

The first definition is quite simple, because our purpose to electrolyze the whiteboard is to help people doing their work easily, more attractive, cost saving, cleaner (because the whiteboard is duster, placement is needed for a huge whiteboard, and the ink difficult to maintain from cleanliness) and so on. The e - board overcomes these requirements with only an application operated in a machine, like the computer or a workstation. The e – board nowadays can be networked into the Internet which make presentation more efficient through the isolation location. These kinds of networked whiteboard, or just net – board (we used net – board to clarify the normal e – board and the networked e - board), however, need some improvisation which voice integration with real – time (RT) architecture have to implement whit it. The aim of voice integration is to satisfy the minimum requirement similar to the real world whiteboard presentation.





Problem also occur within this type of net – board when multiple user access or using it at the same time. Just imagine if there have 5 person login to this application, and they drawing on the whiteboard on this net - board at the same time, or one is drawing and the other one (said intruder) erasing it at the same time, how can this application function. How about if there have more than 100 person login at the same time? (Because the application may integrate through a huge network, like the Internet) So, to overcome this, the application should have the concurrency and security control. This is different with the real world whiteboard and the normal e – board because the real world whiteboard and normal e – board did not contain these types of problems.

Besides, the e – boards nowadays still need some improvisation. There are many applications with their complex structure make them difficult to learn, to control and maintain. Experience users also will meet some difficulties especially when face with a novice of that application (on net – board like the NetMeeting).

A better throughput or respond time is important and must be the one with the high priority in the development. By the way, there have trade off between throughput and security, which mean if we need a better security application, the system throughput will decrease



by the same time. So, there is a requirement for a flexible security where encryption technology can be enable or disable, depend on different situations for example a click by the users or an event to develop a well - define and well - functioning e - board. Our enable or disable it.

objective of this project tend to be:

Current net - board also meet a critical problem where large traffic need to transfer graphic stream (create by drawing on the board) through the network. A sizeable board but controllable (by the server) is intended for a flexible application. A flexible e - board also intent to be integrated by attractive GUI and some graphic effects. These effects perhaps to not over load the system and network traffic.

We now make a simple conclusion on problem definition discussed above as follow:

1. The e - board aim to make jobs easily
2. Voice integration with RT - architecture and some graphic effects is needed
3. A better concurrency control and security control is needed
4. Current e - board is difficult to use: need simple but offer the complete function of whiteboard application
5. Current e - board lack of a sizeable and flexible board





## 1.3 Project Objective

As the problems shown in the problems definition make the intensive following function:  
to develop a well – define and well – functioning e – board. Our objective of this project tend to be:

1. Observe and research some off – the – shelf functioning e-board. Observe their functions, attributes and how the designer organize this
2. Compare several different application and point out their good and leak which at last be our reference on this project
3. Discuss and perhaps meet the improvisation on the leak by the way still maintain the good
4. Group discussion to organize a good teamwork, people management and interactivity between users, administrator and the developer
5. Exposure ourselves to the real development
6. Observe some development functioning methodologies and choose the adequate one to be our methodology



7. Develop a reliable, maintainable, efficiency, attractive, interactive and flexible whiteboard application with the following function:

- Online drawing functions
- Collaborative functions
- RT audio conferencing
- System management
- Easy to use and maintain – user friendly GUI

8. To introduce the use of some network and graphic programming.

9. Finally, at the end of this project, make the students be independence in organizing and managing their project or their developed application

## 1.4 Project Scope

Project scope simply means what can the project do, estimation on how many time is needed to develop the system, money to be spend, people involved and so on. We simply reduce the definition of the





project scope on the modules of this project and ignore the others definitions to clarify our priority on the functioning modules. (This not means the others is not important, but our R and D is prioritize on the system modules)

The whiteboard application is divide into three major parts: Board, Sound and the network architecture. This paper is about the Board, and the others will with my two other partner. As the name specified, the Board is a board or container or pane used to paint, draw, and paste image. Besides, we have integrated some modules on this Board as our project scope.

The whiteboard application develop tends to function the following modules:

### *General*

The Board should function akin to the real whiteboard where painting, imaging and pointing modules are the general.

### *- Painting*

Painting module can divide to three sub modules: Drawing module, Typing module and Coloring module.



### - Drawing

Drawing is the most general purpose of a whiteboard. We can draw anything on the board for our presentation to our audience. We can draw pictures, write, and direct/ focus/ highlight easily using an erasable marker pen on a whiteboard and this apply to our implementation of the Board. There have a word say: a drawing can be present better than hundred of sentences. So, drawing is an important feature that is no exception on this board.

### - Typing

Typing is another alternative for writing where a manageable, sequence but limited character presented on the Board. Sequential and standardize character will clarify the presentation. Data transferring character string is much easier and faster than graphic stream but difficult to integration. Our Board is intensively to integrate this module.

### - Coloring

On a real whiteboard, we can used several type of coloring erasable marker pen for clarify our presentation. We can use different color to highlight different level of important part. Different color also can be use to identify different users.





### - Imaging Effects

Besides painting, imaging also is declared as an important feature, where some image can be paste on the board for references. We can paste some pictures, figures and maybe statistic graphs on the Board for clarify our presentation.

### - Animation Effects

#### - Pointing

The third general feature that equally important is the pointing function. Concurrency control is needed for this module while only one user can point at one time instead of multiple users pointing at the same time. This pointing feature is function only on presentation section. When the pointing is functioning, only the user who is activate the pointing feature can move the mouse cursor, while the other will disable

### Graphic and animations effects

The whiteboard application also intensively be integrated with some graphic effects and animation effects. This module will organize as plug – in features to the board. The idea of this plug – in is to make the program modifiable.



## *- Graphic Effects*

Graphic effect is the effect that changes the graphic's layout, for example the gray scale to change a picture to become white and black, a pencil draw effect, and the darkness effect.

## *- Animation Effects*

Animation effect is adopted similar to graphic with some movement of graphic items within the Board. Examples of animation is like the fly from left effect, suddenly appear, laser and more.

## *GUI*

A nice Graphical User Interface or simply GUI makes the user more comfortable on using this application. Our intensive is to give the users to choose the GUI or the skin themselves. New skin can also be design at future and add to this Board. This enhancement Board feature is intensively to be developed.

## *Filing*

The 'paint' on the board can be save as a bitmap file for future review.





## 1.5 Project Limitation

Due to the imperfect network situation, the whole system will be restricted to some limitation like follow:

- Networking environment - VLAN
- Programming Languages (software) – Visual Basic
- Middleware – depends on software
- Platforms
  - Hardware
  - Operating System – Windows based

## 1.6 Summary

The main purpose of this project is to make R & D on some pre - build whiteboard applications and do some enhancement on functions with some new features added. The expected outcome is an application that fulfills the users' satisfactions.



## CHAPTER 2: LITERATURE REVIEW

### 2.1 Purpose

Literature Review is a background study about the knowledge and information gained in developing this whiteboard system project. The purpose of having this literature review is to have a better understanding about the development tools used to develop this project. Besides that, having this literature review can help to obtain a better knowledge and skills on the development methodologies used while developing a project.

Moreover, we are able to make comparison and study on the developed projects about the strength, advantages and disadvantages of their projects. This indirectly can help the developers to study the weakness of their projects, get to know on how to improve the weakness and fulfill the requirements needed.

Our literature review will cover the following topics but because we are in teamwork, we separate our review for each member. For my path, I will cover the information gather and sources, methodologies and some development software, where else the others covered by my two partners.

#### ➤ Information gather and sources





➤ Development Models (Methodologies)

➤ Audio technologies

➤ Application research (e.g. Netmeeting)

➤ Development software (VB, Java...)

➤ Collaborative technologies

➤ Distributed system & security technologies

➤ Protocol (TCP/IP, UDP, RTP)

## 2.2 Information gather and source

A system is a collection of related parts treated as a unit where its system interacts. It is also a regularly interacting or independent group of elements forming a unified whole. Thus different systems can be developed in different methods and ways. A lot of information need to be gathered about the system that to be developed, the procedures and methodologies used and involved in developing the system.

There are several of sources that can be used in order to gather the relevant information and basically different sources will yield



different kind of information and facts. For example information obtain from books or journals are different from the information obtain from the Internet. If Internet is used to find an information, the keywords or phrases that are being used to search information will yield various sites and some of the sites are totally different from each other and

As for the whiteboard application project, a lot of researches have Some are not related to the information that is being search at all. Besides that, it is also depends on how the research has been carried out. The sources that can be used to obtain information are from the computer programs, system users, procedure manuals and report, documents and forms.

Information gather from the system users can be divided into a number of ways. One of the most popular ways is through the usage of questionnaires. Interview is another way to obtain useful information while the third way is through observation of users activities and behaviors. Computer programs are use to obtain information about the details and flows of data structures or processes.





2.3 Procedure manuals are used to specify user activities in a

business process while forms and documents are very useful sources

### 2.3.1 System Engineering

to gather information such as system data flows and transactions.

Reports are used to indicate the kind of output needed by users.

As for the whiteboard application project, a lot of researches until the end of its useful life. Software engineering is more than just have been made through the Internet on some of the e - boards to programming; the software engineering process generally starts long study the design, features, functionality and differences of this e - before a line of code is written and continues long after the initial boards.

Besides that, the Internet has been used to gather information from various sites on software development tools to be used to develop this project. Moreover, some researches have been done on programming skills at some of the relevant sites via the Internet.

Apart from that, information has been gathered using interview with the presenter, user and the audience regarding the information needed in the application.



## **2.3 Development Models**

### **2.3.1 System Engineering**

System engineering is the application of scientific principles to (1) the orderly transformation of a problem into a working software solution and (2) the subsequent maintenance of that software until the end of its useful life. Software engineering is more than just programming; the software engineering process generally starts long before a line of code is written and continues long after the initial version of the program, has been completed. People and projects following an engineered approach to software development generally pass through a series of phases, or stages.

### **2.3.2 Waterfall Model**

The waterfall model is a systematic sequential approach to software development modeled after a conventional engineering cycle. One phase is completed before the next is entered. There are five phases to the waterfall model, which are:

#### *i. Software Requirements Analysis*

Software requirements analysis defines functional capabilities, performance, design constraints and system





interfaces; also called functional description, functional requirements, and specification. It provides the software designer with representation of information and function that can be translated to data, architectural and procedural design. (Pressman p. 25) "The requirements analysis task is a process of discovery, refinement, modeling, and specification. The software scope, initially established by the system engineer and refined during software project planning, is refined in detail. Models of the required information and control flow, operational behavior, and data content are created. Alternative solutions are analyzed and allocated to various software elements." (Pressman p. 173)

Five areas of effort:

1. Problem recognition
2. Evaluation and synthesis
3. Modeling
4. Specification
5. Review



The information needed can be gathered anyway with any method. This information will then be analysed and form a Software Requirement Specification (SRS). Software requirements phase is very important for analyzing the software problem at hand and concludes with a complete specification of the desired external behavior of the software system to be built. Some persons also call this phase as functional description or functional requirements. The below hypothesis show that why are requirements so important:

1. The later in the development life cycle that a software error is detected the more expensive it will be to repair. The mean of expensive here is the relative cost, time and manpower of repair.
2. Many errors remain latent and are not detected until well after the stage at which they are made
3. In reality development, there are many requirements errors being made
4. Errors made in requirements specifications are typically incorrect facts, omissions, inconsistencies, and ambiguities and can be easily detected





ii. **Software Design**

Software design is translating the requirements into a representation of software that can be assessed for quality before coding begins. There are two phases of software design:

6. Preliminary Design (transforms requirements into architecture; also called specifications, high – level design, architecture design, and functional design)
7. Detailed Design (defines and documents algorithms for each module in the design tree that will be realized as code; also called program design.)

iii. **Coding**

Transforms algorithms defined during the detailed design stage into a computer – understandable language. This is usually performed in two steps: converting the algorithms into a high – level language and converting the high – level language into a machine language; also called programming.

iv. **Testing**



Testing is exercising the software to uncover errors and ensure the system meets defined requirements.

8. Unit testing (checks each coded module for the presence of bugs. Unit testing is aim to ensure that each as – built module behaves according to its specification defined during detailed design; also called module testing and functional testing.)

9. Integration testing (interconnects sets of previously tested modules to ensure that the sets behave as well as they did as independently tested modules; also called string testing and computer software component testing)

10. System Testing (checks that the entire software system embedded in its actual hardware environment behaves according to the SRS)

#### v. *Maintenance*

Maintenance is making adaptation of the software for external changes (requirements changes or enhancements) and internal changes (fixing bugs). When changes are made during





the maintenance phase all preceding steps of the model must be revisited. There are three types of maintenance:

1. Corrective (Fixing bugs/errors)
2. Adaptive (Updates due to environment changes)
3. Perfective (Enhancements, requirements changes)

The problems with the waterfall method are:

- Real projects rarely flow in a sequential process.
- It is difficult to define all requirements at the beginning of a project.

### 2.3.3 Prototype Model

- This model has problems adapting to change.
- A working version of the system is not seen until late in the project's life.

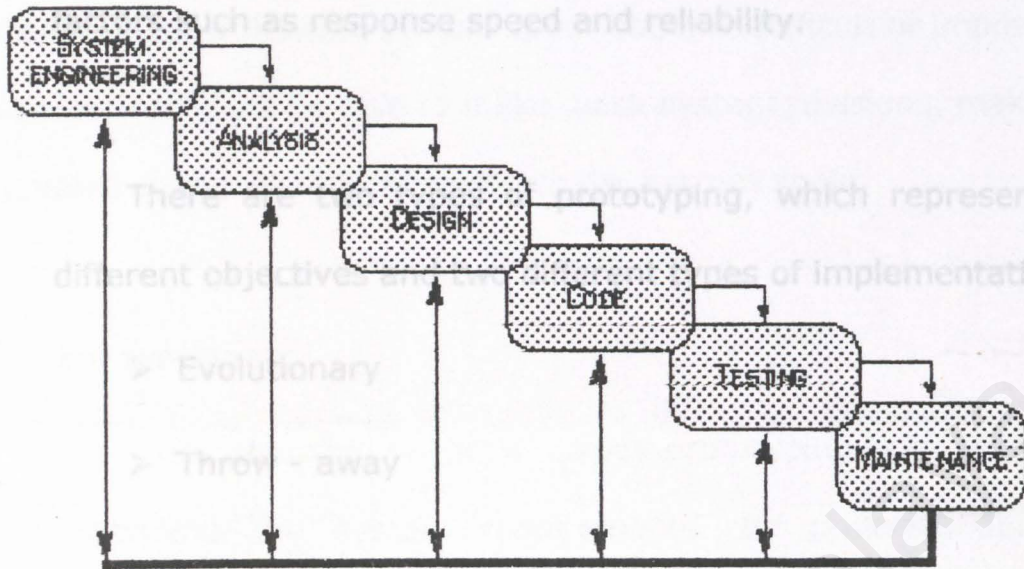


Figure 2.1: The Waterfall Model of software development

### 2.3.3 Prototype Model

Prototyping consists of developing an initial 'model', providing the prototype to the intended users, gathering feedback from the users, and including any revisions or refinements. This process continues until the required system is developed.

Specification, development, and validation activities occur at the same time, since rapid development is important for prototype systems. To deliver a prototype quickly, you may have





to leave out some system functionality or relax non-functional factors such as response speed and reliability.

There are two types of prototyping, which represent two different objectives and two different types of implementation:

- Evolutionary
- Throw - away

#### **Evolutionary:**

The objective of evolutionary prototyping is to deliver a working system to end-users. Evolutionary prototyping starts with the parts of the system, which are clearly understood by the customer/end-user. Adding new features/requirements as they are discovered or proposed by the customer, until a final system is delivered continually develops the system. The evolutionary approach helps anticipate how end-users will use new software systems. The user is given a system, which is unfinished, and then the system is modified and fine-tuned as the user requirements become clear.



Evolutionary prototyping is the only practical way to develop systems where a detailed system specification is difficult or impossible to create. You must be able to make quick system revisions, make the suggested changes, and demonstrate the system again.

### **Throw-Away:**

The objective of throw - away prototyping is to validate or determine the system requirements. By understanding the customer's requirements, a better requirements definition for the system can be developed. (A requirements definition is a top-down process, which takes the 'general' system requirements and broadens it out into specifics, which describe the system/environment interactions.) Throw - away prototyping focuses on undefined or unclear portions of the requirements. During implementation, the parts of the system, which are not understood, are developed first. Throw - away prototyping is intended to determine the system specification so that the end - product of the prototype development phase is that specification.

A fixed decision is made to build a Throw - away prototype to help requirements analysis and validation. After evaluation, the prototype is 'thrown-away' and a production-quality system





is built. Throw - away prototyping extends the requirements analysis process with the intention of reducing overall life cycle costs. (The requirements analysis process begins with understanding the domain and ends with the requirements validation.) The principal function of the prototype is to clarify requirements and provide additional information for managers to assess process risks. Components from the prototype may be reused in the production-quality system. However, customers and end-users should not take the Throw - away and make it into a final delivered system due to the following reasons:

- Misunderstanding between software developers and users
- Important system characteristics may have been relaxed during rapid prototype development (security, robustness or reliability).
- User services which are difficult-to-use or unclear may be found and corrected.
- Changes made during prototype development are usually made in an uncontrolled environment. The prototype code may be the only design specification.
- Incomplete and/or inconsistent requirements may be found.
- In a short time, a working system is available to demonstrate the capabilities and usefulness of the application to management; however, this working system is limited.
- The specification for a production quality system can be derived from the prototype.



## Strengths

- By using the prototype process, customers/designers are able to try out a requirement before agreeing to it.
- Users can discover requirements errors or oversights early in the software process. Requirements validation is performed since users are able to experiment with requirements and the system. The requirements validation process consists of seven factors: correctness, consistency, traceable, realistic, needed, verifiable, and completeness.
- Misunderstanding between software developers and users may be identified while the system functions are demonstrated.
- User services which are difficult-to-use or unclear may be found and corrected.
- Incomplete and/or inconsistent requirements may be found.
- In a short time, a working system is available to demonstrate the capabilities and usefulness of the application to management; however, this working system is limited.
- The specification for a production-quality system can be derived from the prototype.





- Prototyping can be viewed as a risk reduction technique. Experiments have shown that prototyping reduces the number of problems connected to requirements specifications and the overall development costs may be lower if a prototype is developed.

### **Weaknesses**

- Progress is difficult to measure since it is not visible. Also, if the systems are developed rapidly, documentation is usually not created to reflect each version of the system.
- The systems may be poorly structured. The constant change is harmful to the software structure.
- Special skills are often required. Small teams of talented and motivated individuals are needed to succeed with prototyping.
- Major technical problems revolve around the need for rapid software development.
- The prototype may not correspond with the way in which the final system is used, especially with Throw- away prototyping.
- Many software project managers are inexperienced in the areas of planning, cost, and estimating a prototyping project.

Change procedures may not be suitable for controlling rapid changes/development.

- Prototype users/evaluators may be pressured by managers to draw quick conclusions concerning the prototype.
- The cost of prototyping represents a large portion of the total development costs. However, effective prototyping can increase the software quality.

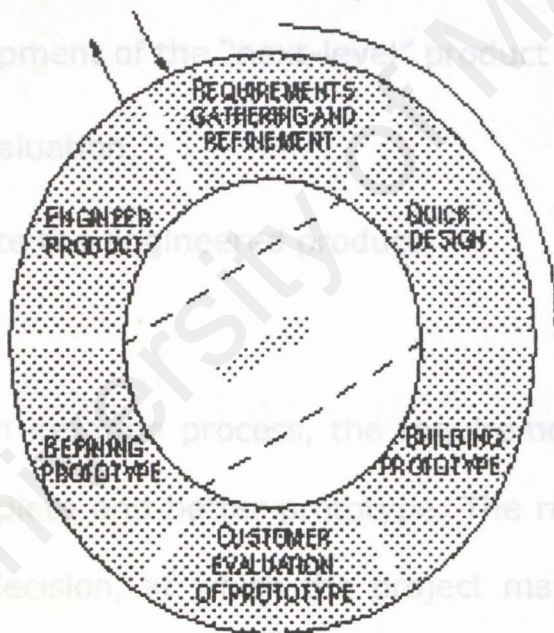


Figure 2.2: The Prototype model of software development

### 2.3.4 The Spiral Model

The Spiral Model has all the elements process in the 2 models mentioned above, with an addition of risk analysis. The model is





presented in a spiral form, in which a circuit around 4 major activities represents each iteration process:

➤ Planning

- Determining objectives and constraints of the project, and define the alternatives

➤ Risk Analysis

- Analysis of alternatives and identification of risks

➤ Engineering

- Development of the "next-level" product

➤ Customer evaluation

- Evaluate the engineered product

## 2.4 Software Developments and Tools

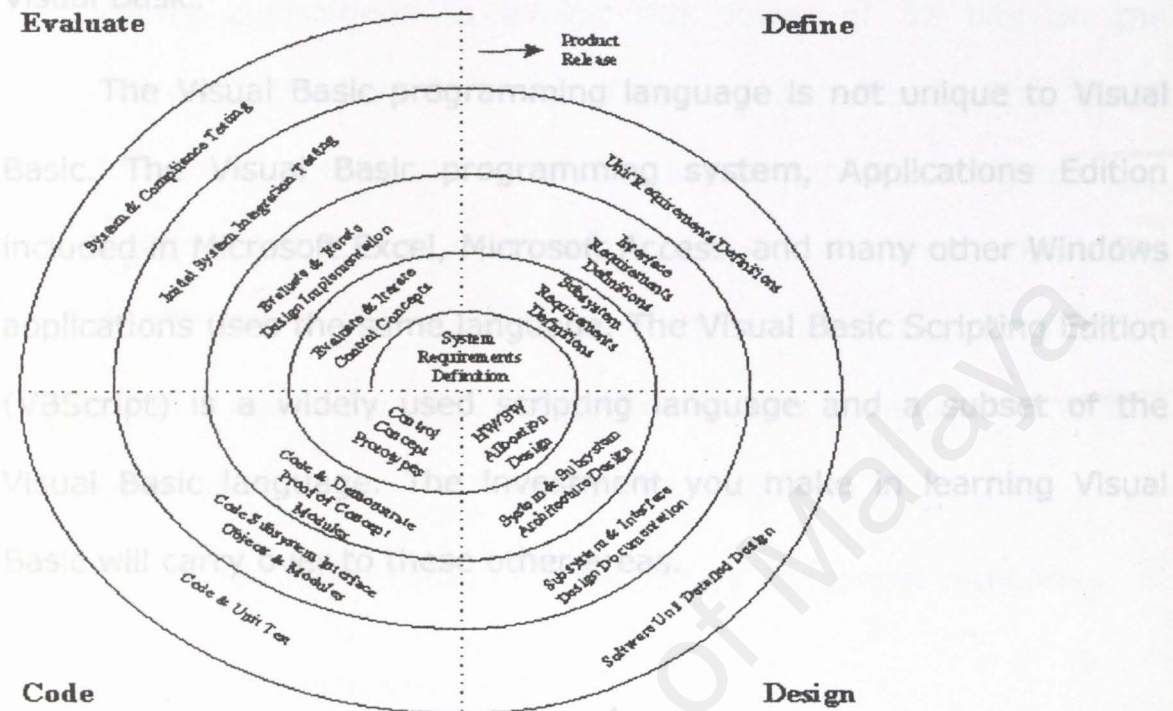
In each iteration process, the requirements are refined to make a more complete and better prototype. The risk analysis always end in an no-go decision, at which the project may be terminated if risks are considered too great.

This model allows the developer to use prototyping approach at any stage in the development, and still manage to maintain the stepwise systematic approach of the life-cycle paradigm.



language, a language used by more programmers than any other language in the history of computing. This is how Microsoft defines Visual Basic.

**Figure 2.3: The Spiral Model of software development**



## 2.4 Software Developments and Tools

### 2.4.1 Visual Basic

The "Visual" part refers to the method used to create the graphical user interface (GUI). Rather than writing numerous lines of code to describe the appearance and location of interface elements, you simply add pre - built objects into place on screen. The "Basic" part refers to the BASIC (Beginners All-Purpose Symbolic Instruction Code)





language, a language used by more programmers than any other language in the history of computing. This is how Microsoft defines Visual Basic.

The Visual Basic programming language is not unique to Visual Basic. The Visual Basic programming system, Applications Edition included in Microsoft Excel, Microsoft Access, and many other Windows applications uses the same language. The Visual Basic Scripting Edition (VBScript) is a widely used scripting language and a subset of the Visual Basic language. The investment you make in learning Visual Basic will carry over to these other areas.

## **Winsock**

Name commonly used for the Windows Sockets programming interface, used to provide a protocol-independent transport interface. A WinSock control allows us to connect to a remote machine and exchange data using either the User Datagram Protocol (UDP) or the Transmission Control Protocol (TCP). Both protocols can be used to create client and server applications. Like the Timer control, the WinSock control doesn't have a visible interface at run time.



## Win32 API

The Microsoft Win32 Application Programming Interface (API) allows applications to exploit the power of 32 bits on the Microsoft Windows® family of operating systems. The Win32 functions, messages, and structures form a consistent and uniform API for all of Microsoft's 32-bit platforms. Using the Win32 API, we can develop applications that run successfully on all platforms while still being able to take advantage of unique features and capabilities of any given platform.

The Win32 API consists of these general functional categories:

- Windows Management
- Graphics Device Interface (GDI)
- System Services
- Multimedia
- Remote Procedure Calls (RPC)

### 2.4.2 Java

The Java programming language is a general-purpose concurrent class-based object-oriented programming language, specifically designed to have as few implementation dependencies as possible. It





allows application developers to write a program once and then be able to run it everywhere on the Internet.

### **Package java.net**

The java.net package provides two basic mechanisms for accessing data and other resources over a network. The fundamental mechanism is called a socket. A socket allows programs to exchange groups of bytes called packets. There are a number of classes in java.net that support sockets, including Socket, ServerSocket, DatagramSocket, DatagramPacket, and MulticastSocket. The java.net package also includes a URL class that provides a higher-level mechanism for accessing and processing data over a network.

### **Package java.awt**

The java.awt package provides the classes that support Java window programming. This package is known as the Abstract Windowing Toolkit. There are a number of classes in java.awt that is appropriate in this application, e.g. the image class, graphics class, color class, event class and so on.



## 2.5 Summary

Generally all the research that has been done was to gain knowledge and information in developing the whiteboard application. Analyses have been done in order to develop this Network Integrated Whiteboard System. In the previous Literature Review section, three software development and tools, and some whiteboard application development models have been studied. After analyzing both models,

The whiteboard samples are not shown here in the literature review, but the others are. We gather our information with several methods, from several targets but in general can be divided into 2 parts, that is the printed resources and the electronic resources. The instances of printed resources are books, journals, and magazines. Meanwhile, the instances of electronic resources are journals on the Internet, e – books, and so on.

We list two methodologies, which we have found and we also include two development softwares or language with some tools, which can be used on fully develop the whiteboard application.





## **CHAPTER 3: METHODOLOGY**

### **3.1 System Analysis**

Analyses have been done in order to develop this Network Integrated Whiteboard System. In the previous Literature Review section, three development models have been studied. After analyzing both models, finally the prototyping model was chosen in developing this project.

are as follow:

#### **3.1.1 Prototyping Development Method**

Prototyping is an approach for establishing a system requirement definition that is characterized by a high degree of iteration, by a very high degree of user participation in the development process and by an extensive use of approach. Through prototyping, the designers can revise forms, input screens, databases, and processing methods, submit them to a limited number of system end users for testing, and revise them for necessary for the final design.

Prototyping is an approach based on an evolutionary view of software development and having an impact on the development process as a whole. It involves producing an early working version (prototypes) of the future application system and experimenting with them.



System prototyping is actually an interactive process that may begin with only a few functions and may start with what that both analysts and users believe a complete set of functions and user requirements that may expand or contract through use and experience.

Once the decision to prototype has been made, there are four main guidelines that must be used when integrating prototyping into the requirements determination phase of the system. The guidelines are as follow:

1. *Work in a manageable modules*

- A manageable module is one that allows users to interact with its key features yet can be built separately from other system modules.

2. *Build the prototype rapidly.*

- Speed is essential to the successful prototyping of an information system.
- Putting together an operational prototype both rapidly and early in the system development allows the analyst to gain valuable insight into how the remainder of the project should go.

3. *Modify the prototype in successful iterations.*





- Construction of the prototype must support modifications. Making the prototype modifiable means creating it in modules that are not highly independent.
- Entering the prototype phase with the idea that the prototype will require modification is a helpful attitude that demonstrates to users how necessary their feedback is if the system is to improve.

#### 4. *Stressing the user interface*

- The user interface with the prototype is very important. For many users, the interface is the system.
- Many of the intricacies of interfaces must be streamlined or ignored altogether in the prototyping phase.

Figure 3.1: Prototyping Model Requirement Analysis

#### 3.1.2 Reason Used Prototyping Model

This prototyping model requirement analysis consists of the following 6 steps as shown in the below figure 3.1. Figure 3.1 show the model for requirement analysis but the over all model will shown by figure 3.2 where different revision will iterate around three type of prototype: requirement prototype, design prototype and system prototype.

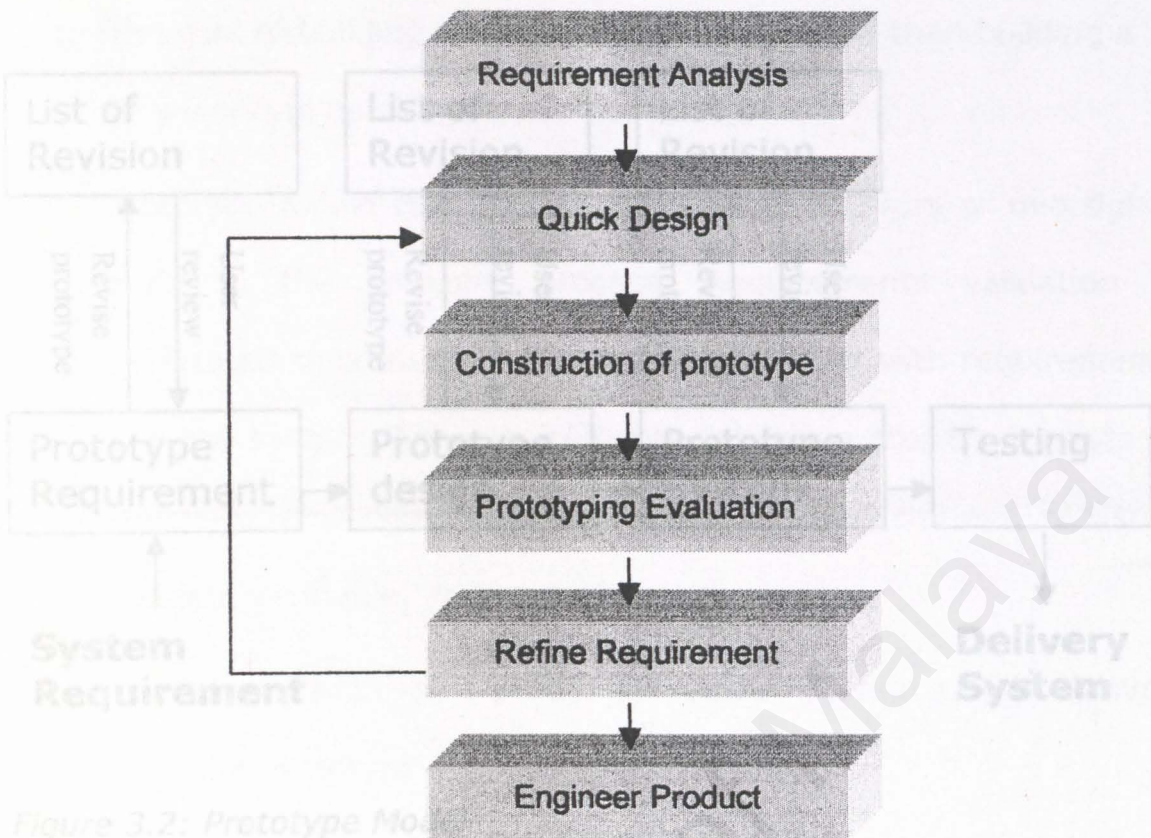


Figure 3.2: Prototype Model

Figure 3.1: Prototyping Model Requirement Analysis

### 3.1.2 Reasons I Used Prototyping Model

The prototyping model was chosen because of the following reasons:

- Prototyping changes the system early in its development because it depends on early and frequent user feedback, which can be used to help modify the system and make it more responsive to actual needs.



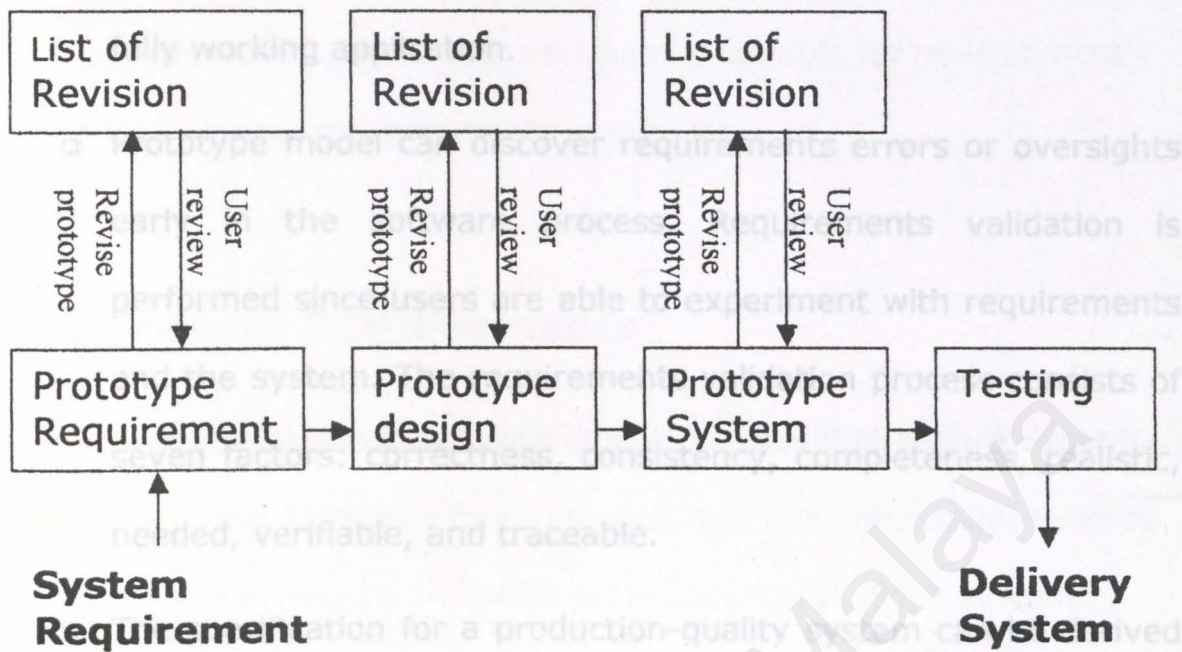


Figure 3.2: Prototype Model

- ❑ Prototyping provides a communication basis for discussions among all the groups involved in the development process, especially between users and developers.
- ❑ Prototyping enables us to adopt an approach to software construction base on experience and experiment.
- ❑ Although it is a working system, it is design in such a way that could be easily changed and enhanced.
- ❑ System prototyping is an interactive process. It may begin with only a few functions later as the development goes on.



- ❑ The cost of building prototypes is relatively less than building a fully working application.
- ❑ Prototype model can discover requirements errors or oversights early in the software process. Requirements validation is performed since users are able to experiment with requirements and the system. The requirements validation process consists of seven factors: correctness, consistency, completeness, realistic, needed, verifiable, and traceable.
- ❑ The specification for a production-quality system can be derived from the prototype.
- ❑ Since Microsoft Visual Basic 6.0 is quite new for us and we only have little knowledge and experience using it, then prototyping is the best method to be used because it can help to reduce errors in the end of the product.
- ❑ Prototyping can be viewed as a risk reduction technique. Experiments have shown that prototyping reduces the number of problems connected to requirements specifications and the overall development costs may be lower if a prototype is developed.





- 3.1.2 Because Visual Basic is RAD tools where modifications can be applied easily, prototype paradigm is suitable for development on this powerful programming language.

## 3.2 System Requirements

Basically, the system requirements for this Whiteboard System are divided into functional requirements, non-functional requirements, hardware requirements and software requirements. Each system requirement is describe below:

### 3.2.1 Functional Requirements

Functional requirements describe an interaction between the system and its environment. The functional requirements in this project are:

- iii. ☐ Whiteboard module
- ☐ Voice Communication module
- iv. ☐ Server module

The description for these three modules will be discussed later on this chapter.



### 3.2.2 Non-Functional Requirements

Non-functional requirements describe the restriction on the system that limits one choice for constructing a solution to the problem. The non-functional requirements for this project are:

- i. *Attractive interface*
  - *Make user comfortable on using this application*
- ii. *Security*
  - *To ensure the program runs smoothly without fear of being corrupted*
  - *To restrict attack by unauthorized users like masquerading, eavesdropping, message tampering, replaying and denial of service with some security techniques like encryption and certificate.*
  - *Password integrated*
- iii. *User friendly*
  - *Easy to use and maintain*
- iv. *Interactive*
  - *Allow better communication between users, server and client*





v. *Efficiency*

- Enable users to retrieve information within a reasonable time

vi. *Customizable*

- Users can customize their interface, change their size, color, style of the drawing pen, change the font type on typing and so on.
- Graphics and animation effects as plug-in

vii. *Reliable*

- The expectation of a system to perform its intended function accurately

3.2.3 Hardware and Software Requirement

The hardware and software requirement of Chat Community System is described as below table.

	Development Environment	Runtime Environment
Hardware Requirement	<ul style="list-style-type: none"><li>• Pentium II 500 Mhz PCs</li><li>• 128 MB RAM</li><li>• Standard input and</li></ul>	<ul style="list-style-type: none"><li>• Pentium II and above.</li><li>• 64MB RAM and</li></ul>



	<p>output devices</p> <ul style="list-style-type: none"><li>• A SVGA Graphic Adapter</li><li>• Sound card</li></ul>	<p>above.</p> <ul style="list-style-type: none"><li>• Standard input and output devices.</li><li>• A SVGA Graphic Adapter</li><li>• Sound card</li><li>• Network setup</li></ul>
<p>Software Requirement</p>	<ul style="list-style-type: none"><li>• Visual Basic 6.0 IDE</li><li>• Ms Office 2000 and Html Help Workshop for documentation</li><li>• Windows 98se/2000 platform</li><li>• Macromedia Flash 5.0</li><li>• Adobe Photoshop 6.0</li></ul>	<ul style="list-style-type: none"><li>• Windows 98/98se/2000</li><li>• Ms Office 2000</li></ul>

Table 3.1: Hardware and software requirements





### 3.3 System Design

System design is a very important step in the system development because it determines the success of a system. Design is the process of transforming the problem into a solution and the description of the solution. Requirements that are found in analysis stage are the one actually translated into design specification. System design will cover the design of system architecture, system modeling and interface design but this will not include the actual coding.

#### 3.3.1 System Architecture Design

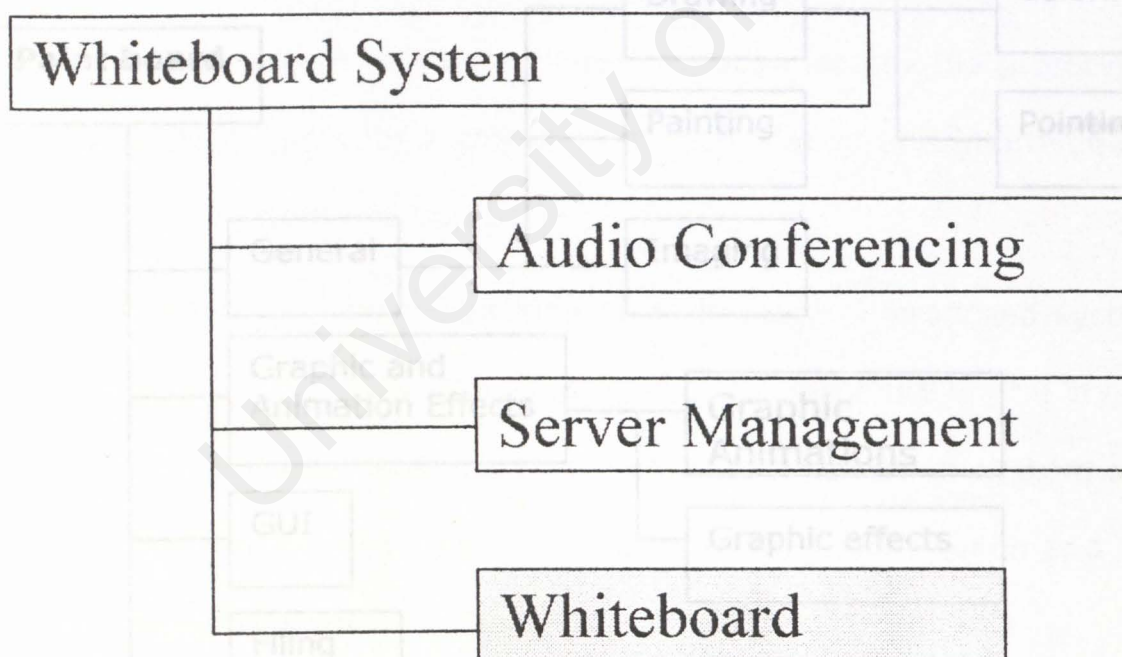


Figure 3.3: system architecture design



Large system normally divided into modules that provide some related set of services. The initial design process of identifying these sub-systems and establish a framework is called architectural design. The system architecture design of the integrated whiteboard system is divide into three parts, which shown in figure 3.4. The whiteboard module is highlighted.

### 3.3.2 Whiteboard module Architecture Design

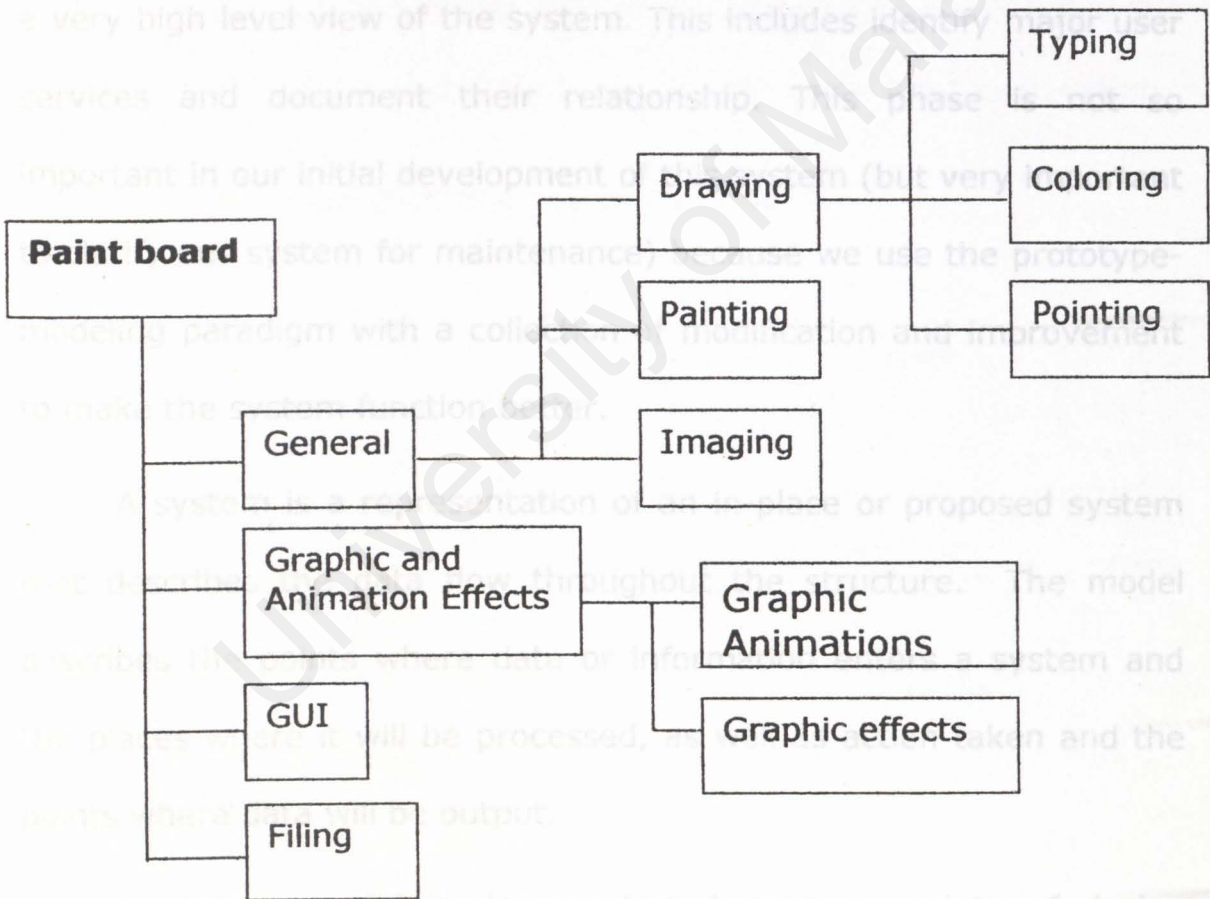


Figure 3.4: The whiteboard module architecture design





As define earlier in chapter one on the project scope, we simplify the project scope with the architecture design on figure 3.4. The other main modules (audio conferencing and server management) are with my two partners. Figure 3.4 is clearly shown the scope and architecture of this system.

### 3.3.3 System Modeling

System modeling is used to create a conceptual of a system, which is a very high level view of the system. This includes identify major user services and document their relationship. This phase is not so important in our initial development of this system (but very important to clarify our system for maintenance) because we use the prototype-modeling paradigm with a collection of modification and improvement to make the system function better.

A system is a representation of an in-place or proposed system that describes the data flow throughout the structure. The model describes the points where data or information enters a system and the places where it will be processed, as well as action taken and the points where data will be output.

A system model is documented through a variety of design diagrams. A design diagram is a graphic or visual representation of a structure. Design diagrams include data flow diagrams (DFD),



structure charts, decision trees and other items. For the whiteboard system, DFD was chosen to represent the system.

The DFD is used as a system-modeling tool because of its great utility. A DFD is a graphic illustration that shows the flow of data and logic within a system.

We will include fraction of our initial model on this system on this report especially the model of the whiteboard module and the main system. Figure 3.5 shows a more detail of the context level diagram. It consists of more detail server mode data flows and the processes involved in handling the data. Some of these processes are logging in, logging out, connection request, connection closed, update active

### 3.3.4 System Functionality Design

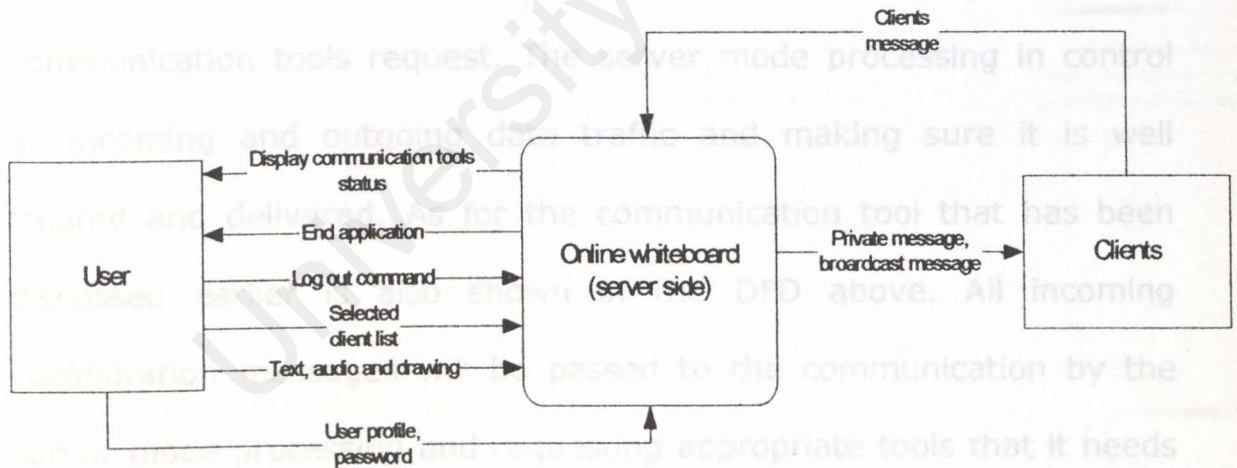


Figure 3.5 Server Mode Context Level Diagram

Figure 3.5 shows a server mode context level DFD of the Whiteboard System. To login to a whiteboard server mode, the user has to





provide a user profile (username) and a group password to the system. After logging into the server mode there are actually three types of input able to be made by a user now on. There are the log out command, selected client list and data (text, audio or drawing). On receiving the client message the server could either response to it using private or broadcast form to handle the sending.

Figure 3.6 shows a more detail of the context level diagram. It consists of more detail server mode data flows and the processes involved in handling the data. Some of these processes are logging in, logging out, connection request, connection closed, update active name list, server mode processing and communication tools request. The core of the entire diagram is the server mode processing and the communication tools request. The server mode processing in control all incoming and outgoing data traffic and making sure it is well treated and delivered. As for the communication tool that has been discussed earlier is also shown in the DFD above. All incoming collaboration messages will be passed to the communication by the server mode processing and requesting appropriate tools that it needs to get the jobs done. Collaboration messages consist of text chat, audio and whiteboard messages.

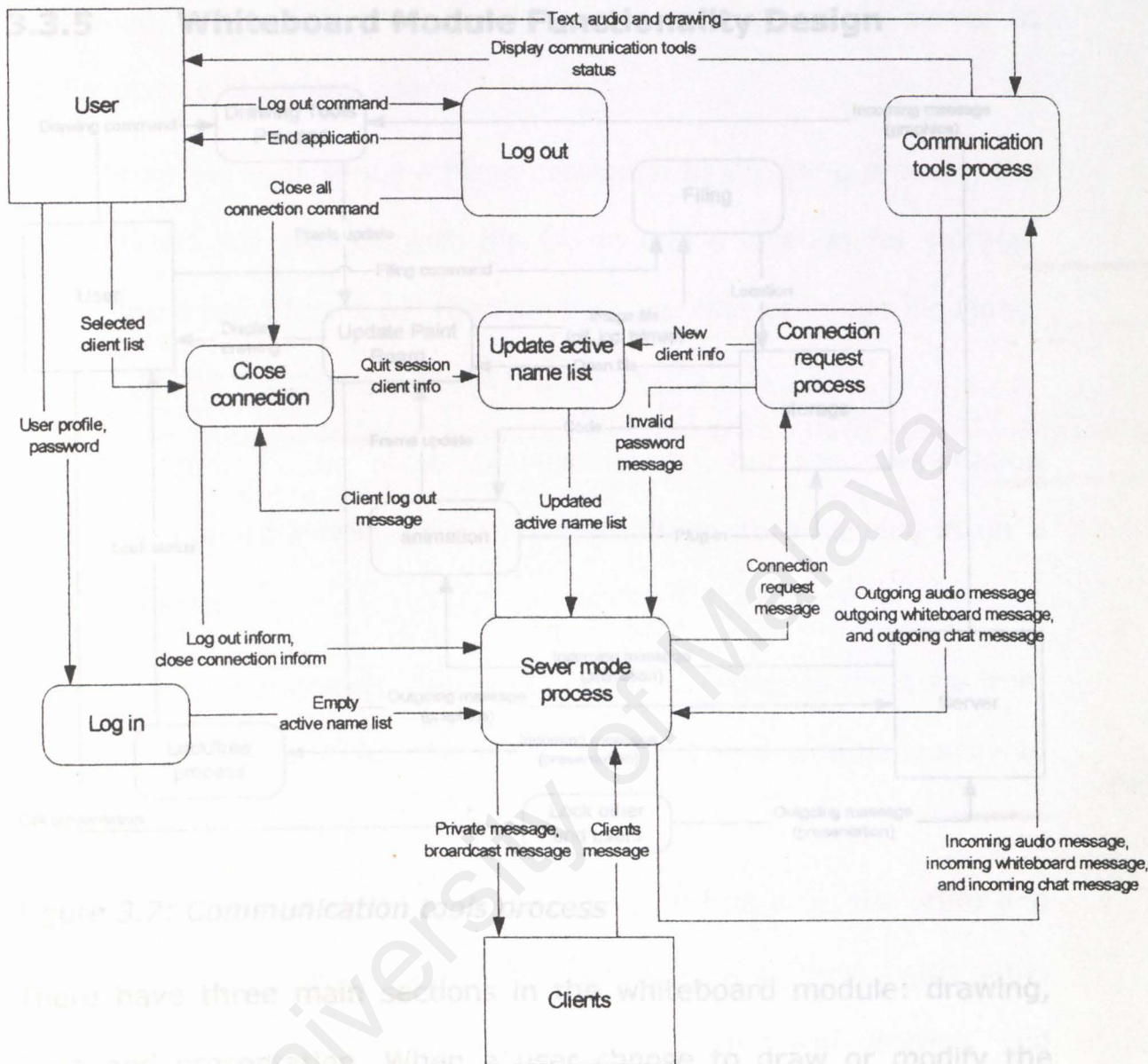


Figure 3.6 Server Mode Level 0 DFD



### 3.3.5 Whiteboard Module Functionality Design

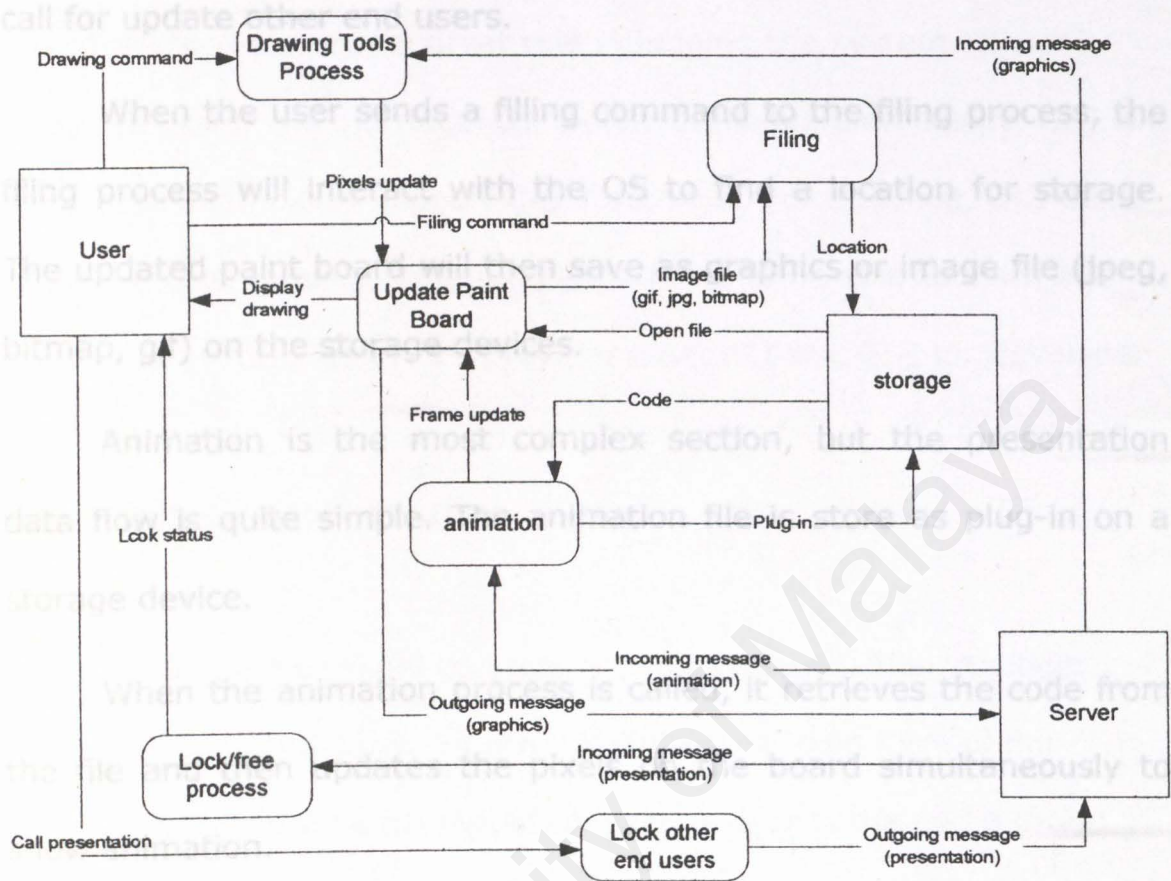


Figure 3.7: Communication tools process

There have three main sections in the whiteboard module: drawing, filing and presentation. When a user choose to draw or modify the whiteboard, he/she send a drawing command to drawing tools process, this process will then update the board will a new pixels behaviors, which then display to the user through the monitor. When interact with server, the steps are almost the same. The different is just that the drawing command is change to graphic stream and message through the connection between two computers. If the user



is drawing, the update process will send a message to the server to call for update other end users.

When the user sends a filling command to the filing process, the filing process will interact with the OS to find a location for storage. The updated paint board will then save as graphics or image file (jpeg, bitmap, gif) on the storage devices.

Animation is the most complex section, but the presentation data flow is quite simple. The animation file is store as plug-in on a storage device.

When the animation process is called, it retrieves the code from the file and then updates the pixels on the board simultaneously to show animation.

When the user called for presentation, it first locks the other end user in the same group. When the lock process is done (with some concurrent controls), the presenter starts present with drawing and speaking (here will interact with the audio conferencing module). Lock means the others can not drawing and speaking at that time until they have assigned the chances to do so.





### 3.4 User Interface Design

User Interface can play a great role in judging the system efficiency.

Thus the better the interface, the better the system efficiency. GUI is a very good example to support this statement. For instance, GUI helps a user with or without computing experience to be able to learn and use a new system faster. VB provides a set of basic GUI for developer to develop their system and though these set of GUI we develop our first prototype model Interface.

Our first prototype model consists of two forms, which is the login form, and the main form. Figure 3.8 shows the login form of our system. A user will be prompted to insert a user name and a group password. Then the user will have to choose either as either server mode to start the application.

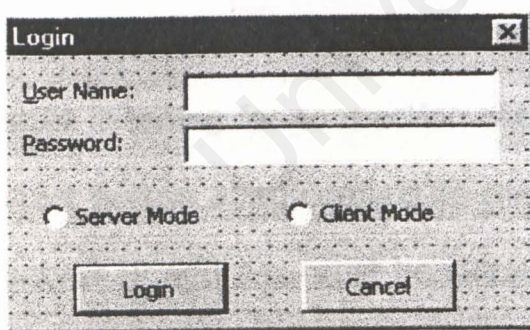


Figure 3.8: Login form

The second form of our prototype model consists of five tabs, which are the Session, Paint, Audio, Chat and Preference.





WhiteBoard System V 1.0 - Not connected

Session Chat Paint Audio Preference

Server Name : myServer Stop Service

Server IP address: 192.168.0.1 Start Service

Server Port : 1234

Local status: Available Remote status: Not connected

Active List  
[Not Connected]

Add To List

Contact List

Figure 3.9: Session Layout

Figure 3.9 shows the first tab of the main form, which is the Session tab and it consists of the all-basic connection setup and status figures. If it is in a server mode, after the user press the start button, it will start listening for connection establishment. While in client mode, the start button as to connect to the dedicated server. Server name or the server IP address must be supply by the client in order to establish connection. The password enter previously in the login form will be used to determined if it is allowed to enter the group or not.



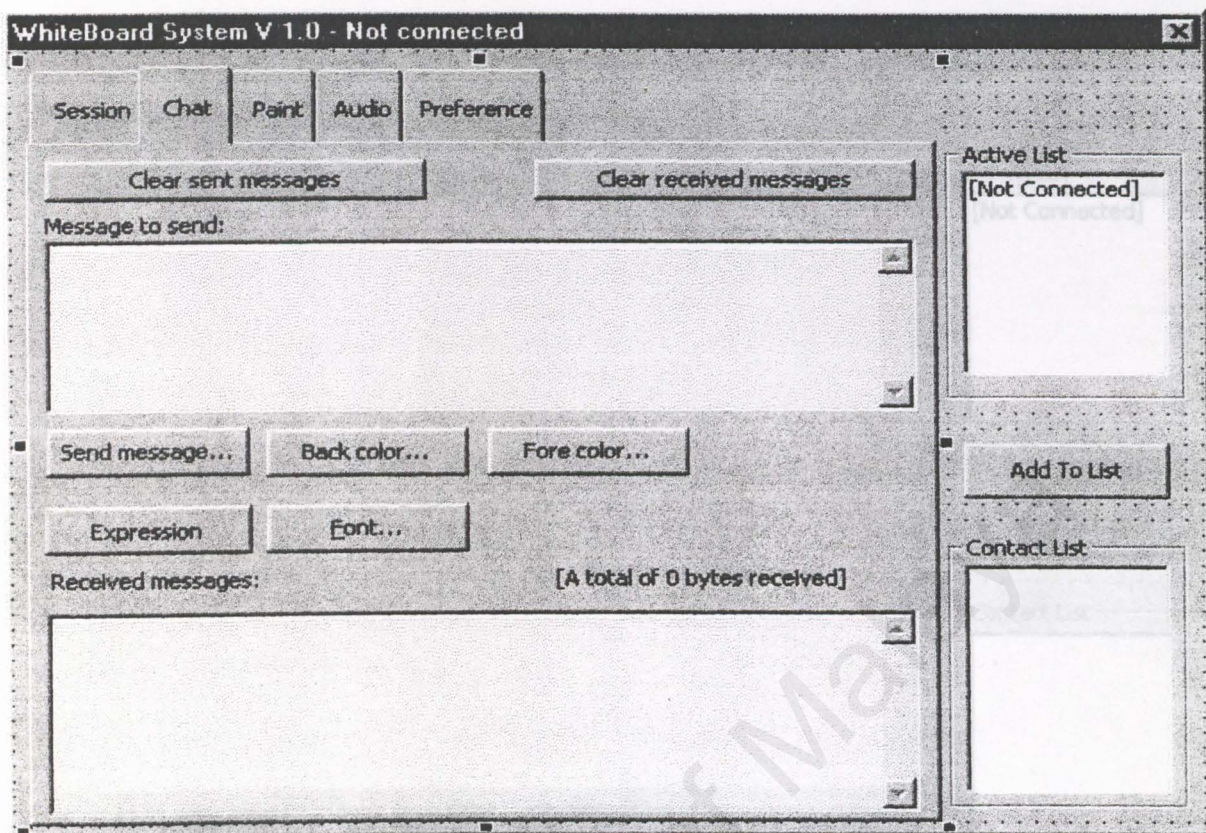


Figure 3.10: Chat Layout

Figure 3.10 shows the second tab of the main form, which is the Chat tab. This is layout consist most of the messaging services that enable user to interact to each other through text. All received message will be put on the *Received messages* box and outgoing message will be put on *Message to send* box. Basic text messaging capability such as show expression, change text font, change text color are also available in this prototype.

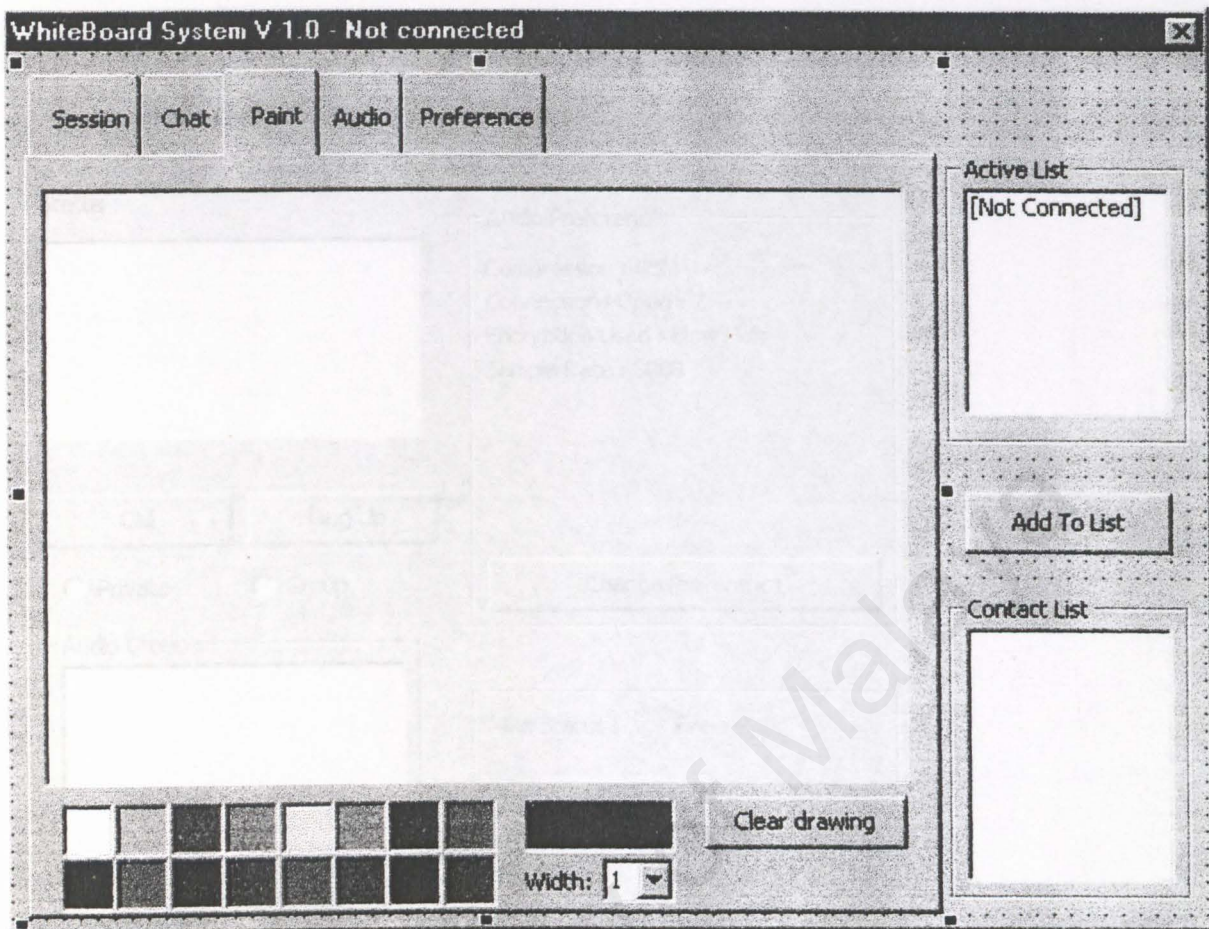


Figure 3.11: Paint Layout

The paint layout consists of a whiteboard. All whiteboard features will be constantly add in to this layout. For the time being, a whiteboard user is only allowed to change the color and size of a pen.

system preference setup and control.



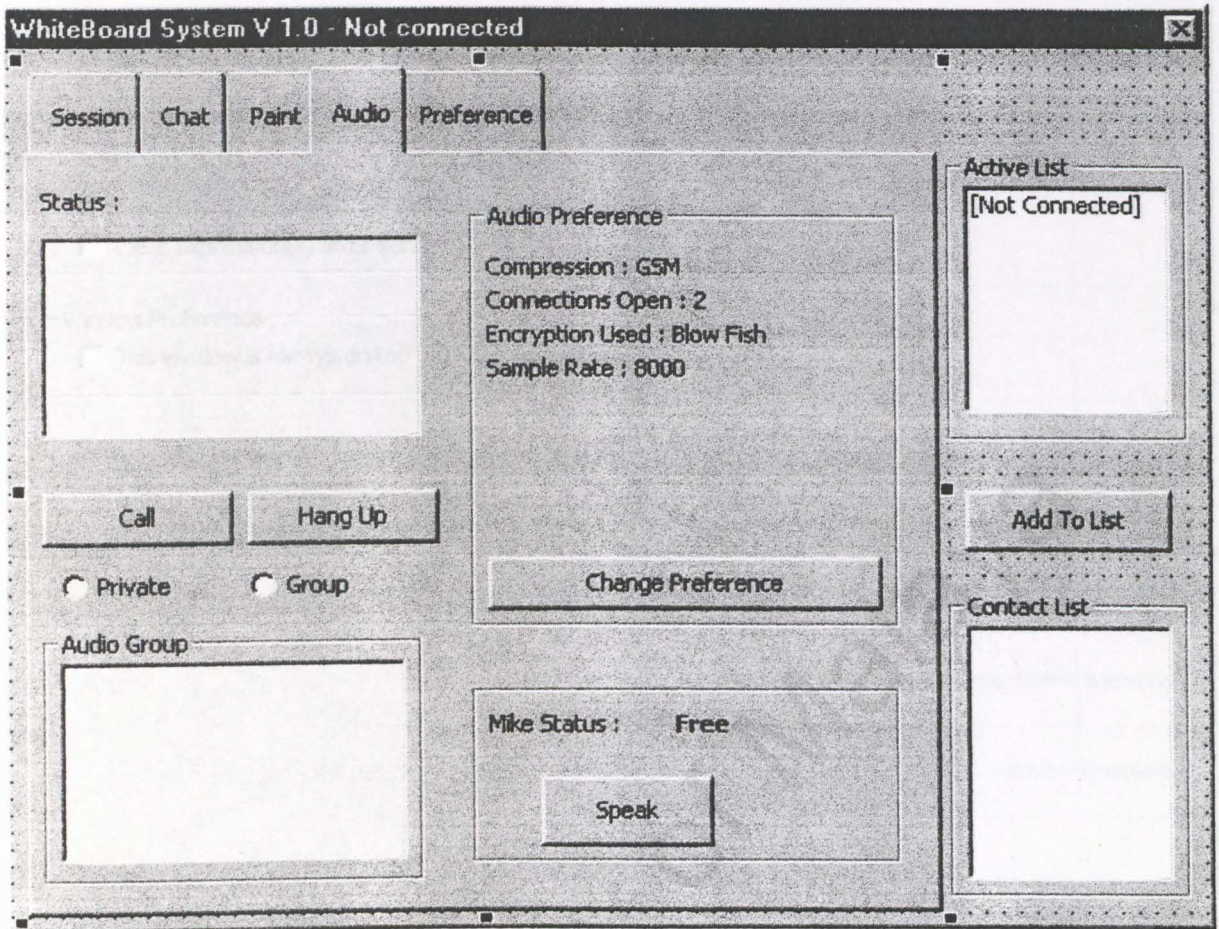


Figure 3.12: Audio Layout

The audio tab as show in Figure 3.12 next to paint tab enables the user to interact with each other through real audio. The last tab, Figure 3.13 stated all basic user preference and it is mainly used for system preference setup and control.



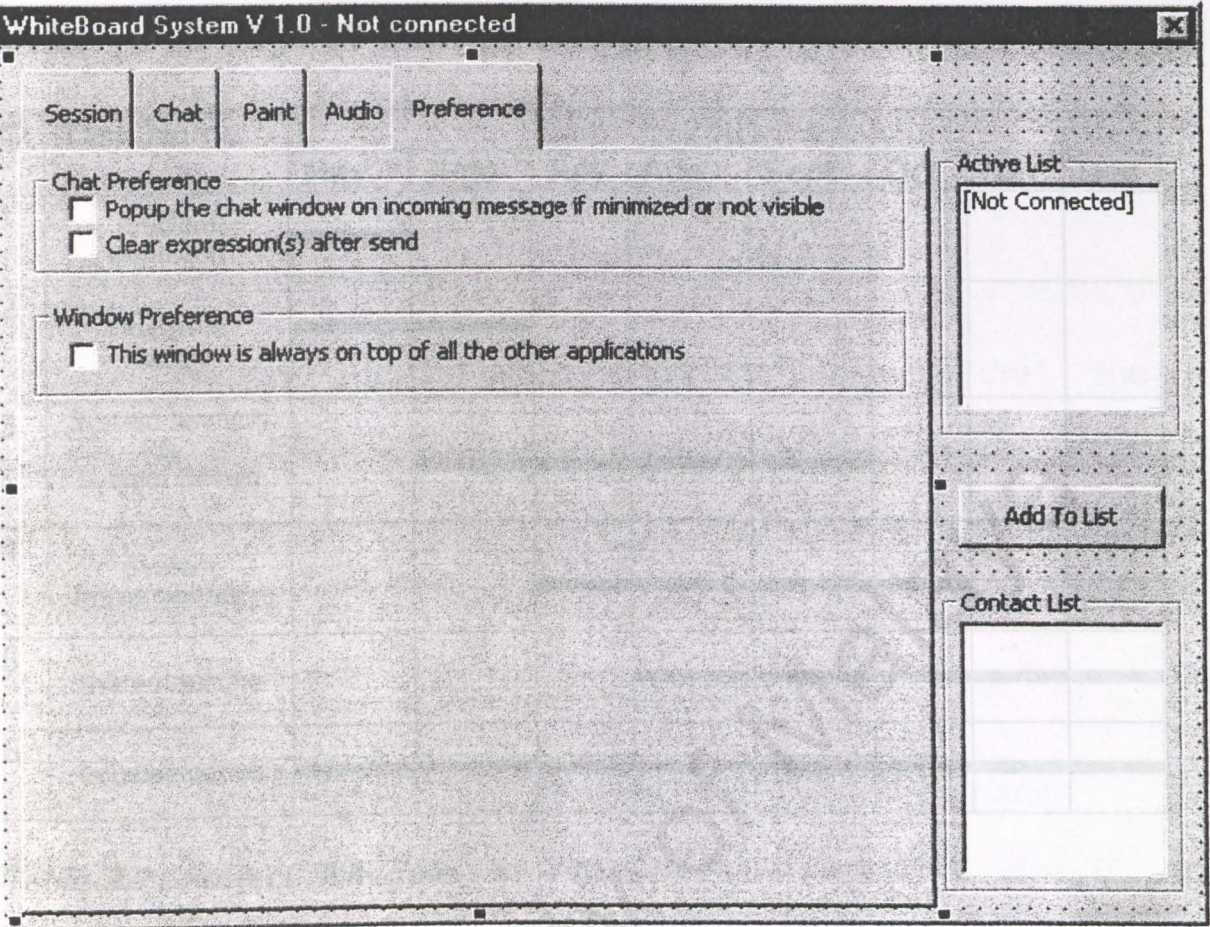


Figure 3.13 User Preference Layout

### 3.6 Statements of Expected Outcome

As we see that there are still a lot of problems for the real time drawing activity, therefore, this application must show some improvement compared to the existing application. The expected outcomes are as below:





### 3.5 Project Scheduling

ID	Task Name	2001							
		May	June	July	Aug	Sept	Oct	Nov	Dec
1	Problem Definition								
2	Literature Review								
3	System Analysis & System Design								
4	System Implementation								
5	System testing								
6	Documentation								

Table 3.2: Project Schedule

### 3.6 Statements of Expected Outcome

As we see that there are still a lot of problems for the real time drawing activity, therefore, this application must show some improvement compared to the existing application. The expected outcomes are as below:



### 1. User friendly.

The application must be easy to use and understand for the users from any levels.

### 2. More attractiveness.

Besides the enough functionality and features, the graphical user interface must be nice and user friendly to increase the attractiveness. Therefore, users will satisfy with the program.

### 3. Easy for enhancement and maintenance.

The coding for the application must be very clear and systematic; hence, the application will become more flexible. If there are any problems, it will be easy to enhance and maintain.

### 4. Smooth and fast enough data transfer.

When we look forward to the problem cause by the time delay of the data transfer, it is possible to design a more effective architecture for this application to solve the problem.

The participants will satisfy with the application only if the data transfer is smooth and fast enough.





## PROJECT DISCUSSION

After we have done lots of research and discussion on our team members, although we have encountered some problems and we tried our best on fact-finding and solution to be taking, we lastly decided to implement our system with a RAD development tools – the Microsoft Visual Basic, which is Windows Platform oriented.

The prototype model is the most suitable methodology to be used on this development of application because of rapid design is very important and no huge database integration. Since we may not build the system on a same environment (we develop the module separately on different environment, but with same resources), the system may be out of expected. Perhaps on the integration phase, this will not rollup a huge problem since some standardization have been perform early. So, a simple clear and standardize prototype is very important and we will put lots of effort on this as the word says: "cost of maintenance on the requirement phase is the most inexpensive"



## CONCLUSION

This research is done in order to adapt us to the real development on R & D, hence capable of develop and deliver an system in future. The main purpose is to develop an online whiteboard system which shown improvement on the off-the-shelf whiteboard application.

Our team have done lots of research on develop this online whiteboard system. This report is about fraction on this online whiteboard system on the whiteboard module, which are the main module of this system. This report includes literature review on the methodologies and development software. Finally, we capable to manage ourselves on develop a system.





## REFERENCES

- I. Gary B. Shelly, Thomas J. Cashman, Judy Adamski, Joseph J. Adamski, 1991. *Systems Analysis and Design*. Boyd & Fraser.
- II. Timothy Ramteke, 1999, *Networks*, Prentice Hall publishing
- III. Jeffrey L. Whitten, Lonnie D. Bentley, Kevin C. Dittman, 2001. *System Analysis and Design Methods*, 5<sup>th</sup> ed, McGraw-Hill Irwin.
- IV. Alan R. Apt, 2001. *Software engineering theory and practice*, 2<sup>nd</sup> ed, Prentice Hall International, inc.
- V. Alan M. Davis, 1993. *Software Requirements Objects, Functions, and States*, Prentice Hall, Inc.
- VI. Roger Jennings, 1999. *Roger Jennings' Database Developer's Guide With Visual Basic 6*, 1<sup>st</sup> ed, SAMS Publishing.
- VII. Steven Roman "Win32 API Programming with visual basic", First Edition, United States of America, O'Reilly & Associates, Inc, 101 Morris street, Sebastopol, CA 95472, January 2000, 511 pages.
- VIII. Microsoft Development Network (MSDN) JAN 2001, <http://www.msdn.microsoft.com/>
- IX. <http://mvps.org/htmlhelpcenter/>
- X. <http://www.mvps.org/vbnet/>
- XI. <http://www.vbweb.co.uk/>



## CHAPTER 4: System Implementation

### 4.1 Standards and procedures

This chapter concerning on the direct correspondence between the program design and the program code components. The entire design process is of little value if the design's modularity is not carried forward into the code. Design characteristic such as low coupling and high cohesion, should also be program characteristics, so that the algorithms, functions, interface and data structure can be traced easily from design to code and back again.

We are using the prototype methodology (explain in chapter 3) for our system implementation so that our program coding is more flexible on development, but still, some standards and procedures should be followed as the program characteristics. Standards and procedures can help avoid mistakes and organize thoughts. The following list some example standard and procedure we used on our implementation:

Standards and procedures	Examples
1. Variables and procedures' name	Const SCK_CODE_CLEAR_DRAW Const SCK_CODE_DISCONNECTED = "[Disconnected]" Const SCK_CODE_JOINED = "[Joined]" Const SCK_CODE_LINE = "[Line]"





4.2	System	Const SCK_CODE_PAINT = "[Paint Picture]"  Const SCK_CODE_STAR = "[Star]"
2. Methods of code documentation	Indentation	For I=0 to 3  If txtNum <> "" Then  txtNum = "On Drawing"  End IF  Next I
3. Code structuring	Nesting, Looping	If mark > 80 Then Skor = 'A' ElseIf mark > 70 Then Skor = 'B' ElseIf mark > 60 Then Skor = 'C' ElseIf mark > 50 Then Skor = 'D' Else Skor = 'E' End If
4. Controls standard	Height properties for text box and label is 285  Text font is "Verdana"  Others	
5. User Manual	We choose to make a help file for documenting our system to help user find out the solutions if problems have met	

Table 4.1: Simplified Class Diagram

## 4.2 System Design

As we follow the prototyping methodology, several revisions have been made and at last, we finalize our system design to make the following implementation of our system. Figure 4.1 shows the finalized DFD for the whiteboard subsystem. Refer to figure 3.2, this is the final prototype design. Figure 4.2 where else define the data flow in the Tools module.

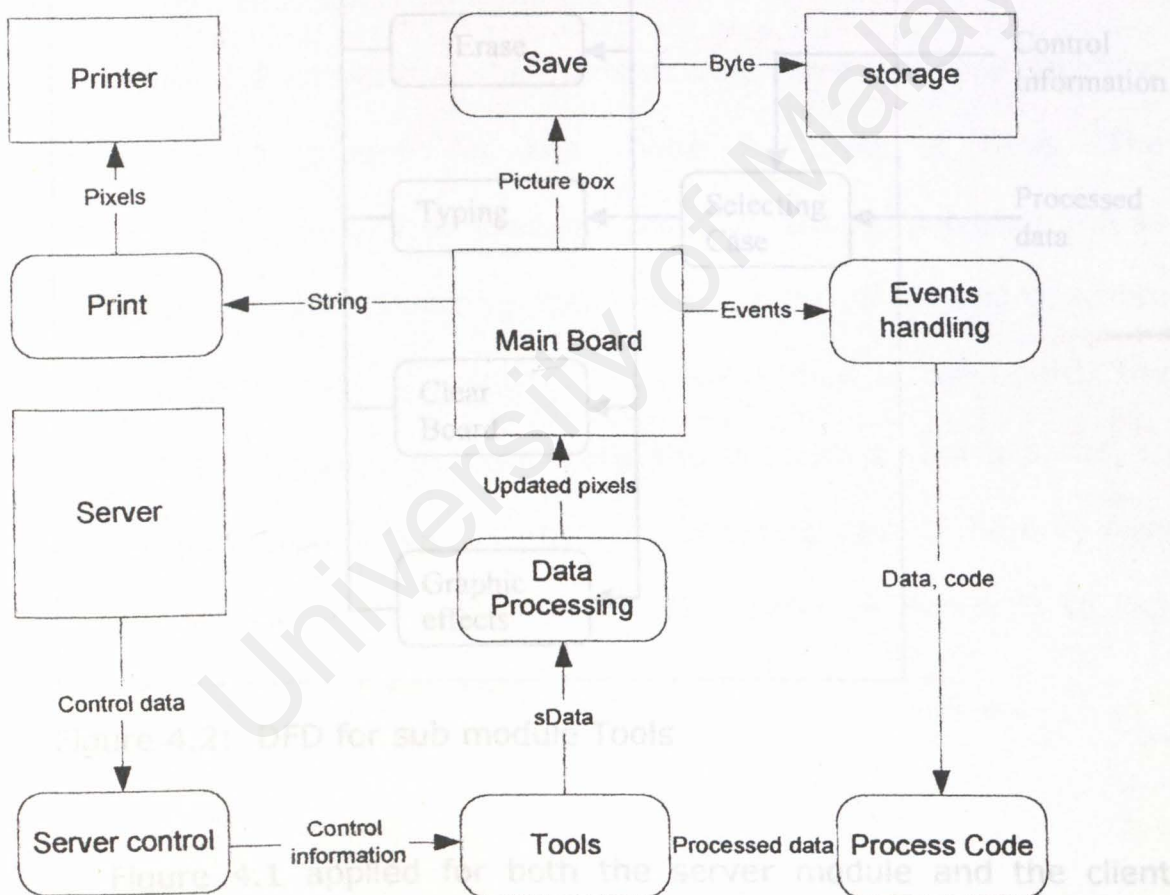


Figure 4.1: Finalize White Board Module DFD



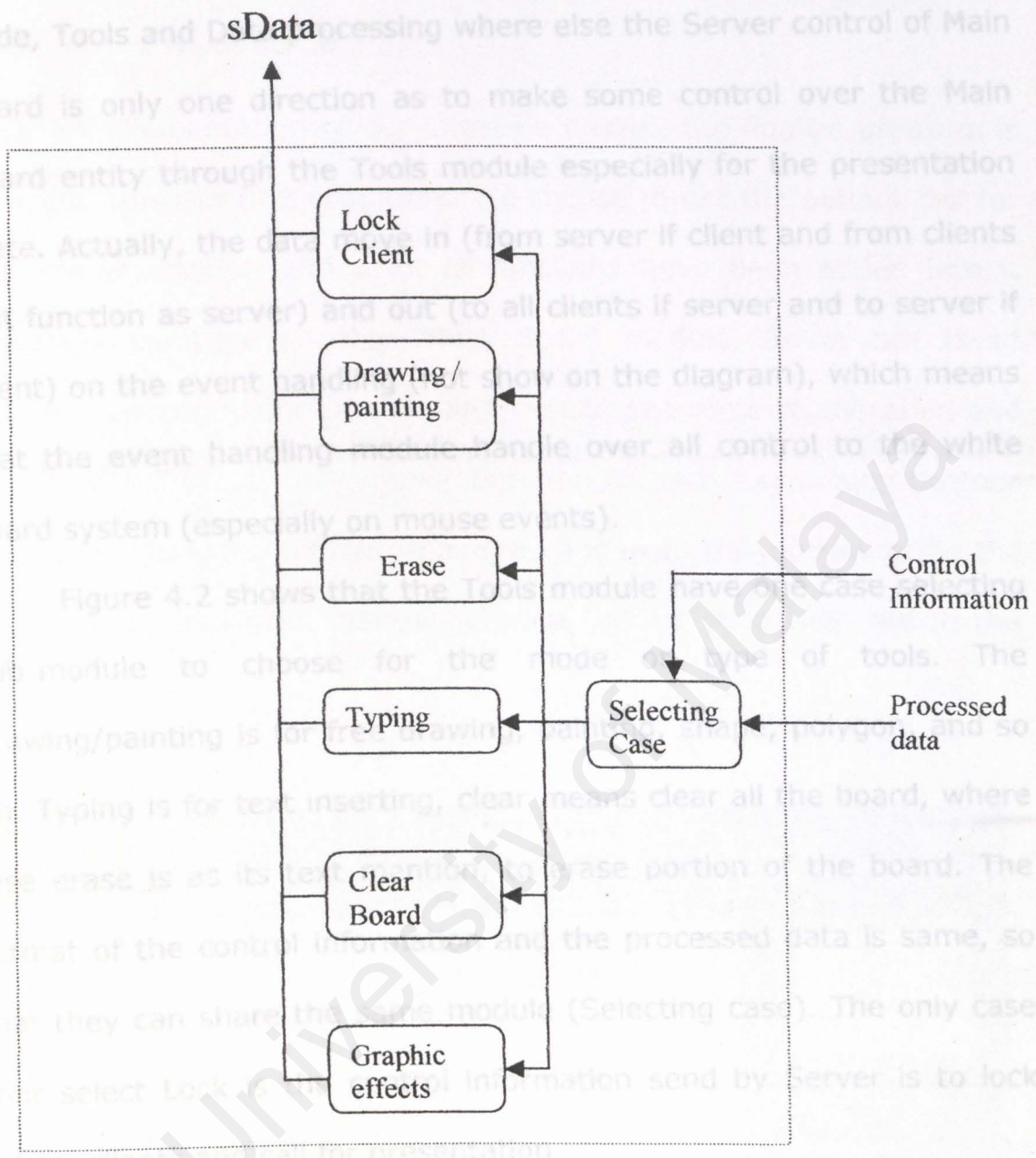


Figure 4.2: DFD for sub module Tools

Figure 4.1 applied for both the server module and the client module for whiteboard. We can simply observe from the figure that there have loops between the "Main board", event handling, Process



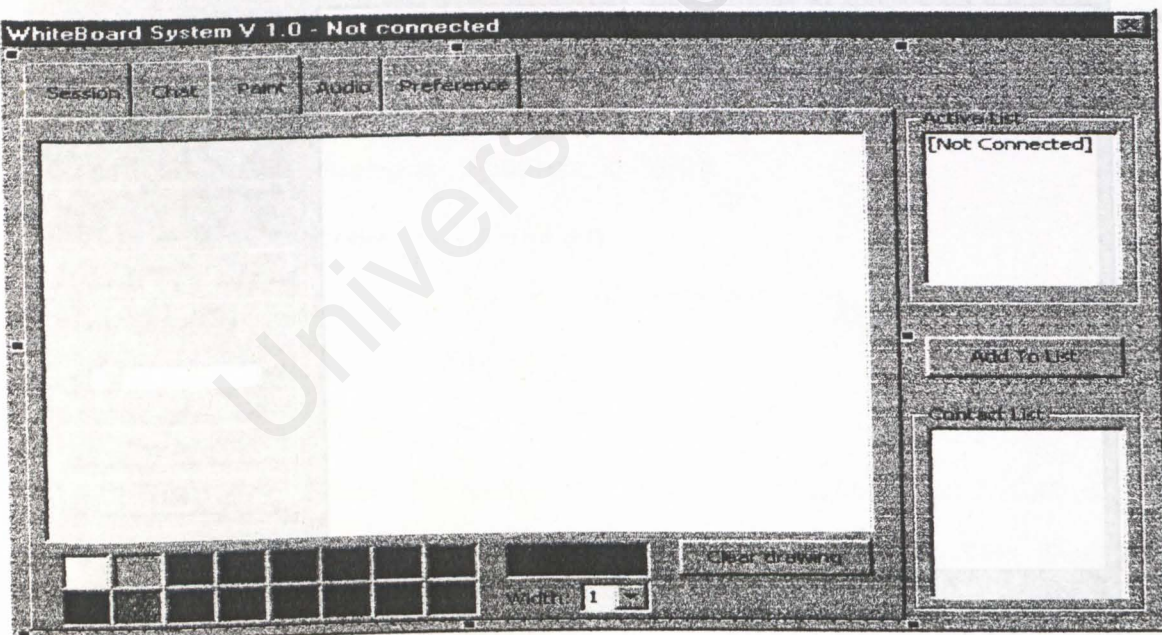
Code, Tools and Data processing where else the Server control of Main Board is only one direction as to make some control over the Main Board entity through the Tools module especially for the presentation state. Actually, the data move in (from server if client and from clients if it function as server) and out (to all clients if server and to server if client) on the event handling (not show on the diagram), which means that the event handling module handle over all control to the white board system (especially on mouse events).

Figure 4.2 shows that the Tools module have one case selecting sub-module to choose for the mode or type of tools. The drawing/painting is for free drawing, painting, shape, polygon, and so on. Typing is for text inserting, clear means clear all the board, where else erase is as its text mention, to erase portion of the board. The format of the control information and the processed data is same, so that they can share the same module (Selecting case). The only case that select Lock is the control information send by Server is to lock others clients and call for presentation.



## 4.3 Interface

On the implementation of our interface design, the finalize program is different from our first prototype. We choose to use the outlook bar for our implementation and a lot of functions have been added into it especially compare on the White Board module. Below are three different version of the module and we can see some modification and enhancement on it. Fully implementation of each functioning buttons and tools (the user manual) has been added to this document as the Appendixes. This user manual will also added as a help file in the system.







Form1

Effects

Name:

Connect to:

Draw:

Connection (Server Mode). These sockets function only

ard module for data sending and receiving. When it is c

a recording or receiving event is signaled, the processes

5025 X 105 Pencil

The White Board

Effects

Control Panel

Draw Width: 3

☐ Presentation

Pencil





At last, we just concentrate on the implementation of the module using Visual Basic. The connection handling made two sockets for White Board module, named *sckConnect* (Client Mode) and *sckConnection* (Server Mode). These sockets function only on White Board module for data sending and receiving. When it is connected and a recording or receiving event is signaled, the processes would be run independently to the system (as we describe above).

Integration Module:	
<b>Module =</b> <b>miscModWB</b>	<p>Global Variables:</p> <p>Double:    sinTab(360), cosTab(360)</p> <p>Integer:    curTool, iIsBmp</p> <p>Long:       FillCol, i, j, curX, curY, firstX, firstY, r, g, b, tCol, lArrCol(), sX, sY, tX, tY, propFillStyle, cuX, cuY, stat, stat1, wX1, wY1, wX, wY</p> <p>Boolean:   IsBitmap, IsCurOK</p> <p>String:     sBmpPath</p> <p>Variants:   StartX, StartY, Marked, OldX, OldY, MoveX, MoveY, NowMove</p>
	<p>Enum :</p> <p>Tools: {   Pencil, Star, HorzLine, VertLine, Cross, DiagCross, DiagLineLR,   DiagLineRL, UdefPoly, InsText, StraightLine, Polygon, Rect, tCircle, FilledRect, FilledCircle, FillRgn, tErase, EfHammer, EfHook }</p>



Module:  frmWBMain	<p>Public Const:</p> <p>Pi = 3.14159265359, BRadius As Integer = 120, FW_BOLD = 700, FW_EXTRABOLD = 800, FW_EXTRALIGHT = 200, FW_HEAVY = 900, FW_LIGHT = 300, FW_MEDIUM = 500, FW_NORMAL = 400, FW_SEMIBOLD = 600, FW_THIN = 100, DEFAULT_CHARSET = 1, OUT_DEFAULT_PRECIS = 0, CLIP_DEFAULT_PRECIS = 0, PROOF_QUALITY = 2, FF_DONTCARE = 0</p>
	<p>API Functions:</p> <p>Kernel32.dll</p> <ul style="list-style-type: none"><li>- GetPrivateProfileString</li><li>- WritePrivateProfileString</li></ul> <p>gdi32.dll</p> <ul style="list-style-type: none"><li>- SetPixel</li><li>- GetPixel</li><li>- ExtFloodFill</li><li>- PolyBezierTo</li><li>- MoveToEx</li></ul>
	<p>Methods:</p> <ul style="list-style-type: none"><li>- RenPanel – update current tools on status bar</li><li>- DrawBCircle – Hammer and Hook Effects</li><li>- PrepPic – set curX to main board width and curY to main board height</li><li>- FileExist - check for file existed</li><li>- IsCurOKFirst – check for cursor</li></ul>





<b>Module:</b>  <b>frmWBMain</b>	<b>Const:</b>  CODE = "[Presentaion code]" SCK_CODE_CLEAR_DRAW = "[Clear Draw]" SCK_CODE_DISCONNECTED = "[Disconnected]" SCK_CODE_JOINED = "[Joined]" SCK_CODE_LINE = "[Line]" SCK_CODE_PAINT = "[Paint Picture]" SCK_CODE_STAR = "[Star]" SCK_CODE_WRITE = "[Typing]" SCK_CODE_RECT = "[Rectangle]" SCK_CODE_FRECT = "[Fill Rectangle]" SCK_CODE_FCIRCLE = "[Fill Circle]" SCK_CODE_CIRCLE = "[Circle]" SCK_CODE_FLIP1 = "[Flip1]" SCK_CODE_FLIP2 = "[Flip2]" SCK_CODE_FLIP3 = "[Flip3]" SCK_CODE_HAMMER = "[Hammer]" SCK_CODE_HOOK = "[Hook]" SCK_CODE_NPOLYGON = "[Normal Polygon]" SCK_CODE_UDPOLYGON = "[User Defined Polygon]" SCK_CODE_PRESENTATION = "[Presentation]" SCK_CODE_PRESENTATION_STOP = "[Stop]"
	<b>Properties :</b>  SckConnect – Client socket  SckConnection – Server socket  MbServer – Desired Mode miNumConnections– Connection Loaded  Buttons – buttons collection  miX, miY, stX, stY



Methods :

- bConnencted
- Clr
- ClearStuff
- SFormatSend
- ProcessData
- FilePrint
- FileSave
- Rect
- SCurrentColor
- SendToAll, SendToServer, SendToPerson, SendToSelf
- SendToAllButOriginator
- SlongParam, sParam1, sParam, sParam
- StartWhiteboardConnect
- StartWhiteboardListen

Table 4.1: Simplified Class Diagram

## 4.4 Sample Coding

The whiteboard module consists of two VB form and one VB modules: the frmWBMain, frmWBPrint and modMiscWB, which contain about 2000 lines of code. We will list down one procedure as sample coding here.

```
Public Sub ProcessData(vsString As String, viConnection As Integer)
```

```
    Dim i As Integer
```

```
    Dim sCommand As String
```

```
    Dim sInstruction As String
```

```
    Dim sData As String
```

```
    Dim bTemp As Boolean
```

```
    Dim iCount As Integer
```

```
    Dim iUser As Integer
```

```
    Dim sFile As String
```

```
    Dim idrawwidth As Integer, iForeColor As Integer
```





```

Do While InStr(1, vsString, vbCrLf)
    sCommand = Mid(vsString, 1, InStr(1, vsString, vbCrLf) - 1)

    sInstruction = Mid(sCommand, 1, InStr(1, sCommand, "]"))
    sData = Mid(sCommand, InStr(1, sCommand, "]") + 1, Len(sCommand))

    Select Case sInstruction
        Case CODE
            If sData = True Then
                picDraw.Enabled = False
                cmdClearDraw.Enabled = False
            Else
                picDraw.Enabled = True
                cmdClearDraw.Enabled = True
            End If
        Case SCK_CODE_CLEAR_DRAW
            picDraw.Cls
            picDraw.Picture = Nothing

            If mbServer Then
                SendToAllButOriginator SCK_CODE_CLEAR_DRAW, viConnection
            End If
        Case SCK_CODE_DISCONNECTED
        Case SCK_CODE_JOINED
            If mbServer Then
                SendToAll SCK_CODE_JOINED & sData, False
            End If
        Case SCK_CODE_LINE
            picDraw.DrawWidth = (sParam1(sData, 6))
            picDraw.Line (sParam(sData, 1), sParam(sData, 2))-(sParam(sData, 3),
sParam(sData, 4)), sParam(sData, 5))

            If mbServer Then
                SendToAllButOriginator SCK_CODE_LINE & sData, viConnection
            End If
            picDraw.DrawWidth = lblPenSize
        Case SCK_CODE_PAINT
            picDraw.Picture = LoadPicture(App.Path & "\" & sData)

            If mbServer Then
                SendToAllButOriginator SCK_CODE_PAINT & sData, viConnection
            End If
        Case SCK_CODE_STAR
    
```



```
picDraw.DrawWidth = (sParam1(sData, 6))
picDraw.Line (sParam(sData, 1), sParam(sData, 2))-(sParam(sData, 3),
sParam(sData, 4)), sParam(sData, 5)
End If
If mbServer Then
    SendToAllButOriginator SCK_CODE_LINE & sData, viConnection
End If
picDraw.DrawWidth = lblPenSize
Case SCK_CODE_WRITE
    If sParam(sData, 5) <> "" Then
        picDraw.CurrentX = sParam(sData, 1)
        picDraw.CurrentY = sParam(sData, 2)
        iForeColor = picDraw.ForeColor
        picDraw.ForeColor = sParam(sData, 3)
        picDraw.Print sParamstring(sData, 5, sParam(sData, 4))
        picDraw.ForeColor = iForeColor
        If mbServer Then
            SendToAllButOriginator SCK_CODE_WRITE & sData, viConnection
        End If
    End If
Case SCK_CODE_FRECT
    idrawwidth = picDraw.DrawWidth
    picDraw.DrawWidth = sParam(sData, 6)
    picDraw.FillStyle = 0
    picDraw.FillColor = (sParam(sData, 5))
    picDraw.Line (sParam(sData, 1), sParam(sData, 2))-(sParam(sData, 3),
sParam(sData, 4)), (sParam(sData, 5)), BF
    picDraw.FillStyle = 1
    SendToAllButOriginator SCK_CODE_RECT & sData, viConnection
End If
If mbServer Then
    SendToAllButOriginator SCK_CODE_FRECT & sData, viConnection
End If
picDraw.DrawWidth = idrawwidth
Case SCK_CODE_FCIRCLE
    idrawwidth = picDraw.DrawWidth
    picDraw.DrawWidth = sParam(sData, 6)
    picDraw.FillStyle = 0
    picDraw.FillColor = (sParam(sData, 5))
    r = Sqr((sParam(sData, 1) - sParam(sData, 3)) * (sParam(sData, 1) -
sParam(sData, 3)) + (sParam(sData, 2) - sParam(sData, 4)) * (sParam(sData, 2) -
sParam(sData, 4)))
    picDraw.Circle (sParam(sData, 1), sParam(sData, 2)), r, (sParam(sData, 5))
    picDraw.FillStyle = 1
```





If mbServer Then

SendToAllButOriginator SCK\_CODE\_FCIRCLE & sData, viConnection

End If

picDraw.DrawWidth = idrawwidth

Case SCK\_CODE\_CIRCLE

idrawwidth = picDraw.DrawWidth

picDraw.DrawWidth = sParam(sData, 6)

picDraw.FillStyle = 1

$r = \text{Sqr}((\text{sParam}(\text{sData}, 1) - \text{sParam}(\text{sData}, 3)) * (\text{sParam}(\text{sData}, 1) -$

$\text{sParam}(\text{sData}, 3)) + (\text{sParam}(\text{sData}, 2) - \text{sParam}(\text{sData}, 4)) * (\text{sParam}(\text{sData}, 2) -$

$\text{sParam}(\text{sData}, 4)))$

picDraw.Circle (sParam(sData, 1), sParam(sData, 2)), r, (sParam(sData, 5))

If mbServer Then

SendToAllButOriginator SCK\_CODE\_CIRCLE & sData, viConnection

End If

picDraw.DrawWidth = idrawwidth

Case SCK\_CODE\_RECT

idrawwidth = picDraw.DrawWidth

picDraw.DrawWidth = sParam(sData, 6)

picDraw.FillStyle = 1

picDraw.Line (sParam(sData, 1), sParam(sData, 2))-(sParam(sData, 3),

sParam(sData, 4)), (sParam(sData, 5)), B

If mbServer Then

SendToAllButOriginator SCK\_CODE\_RECT & sData, viConnection

End If

picDraw.DrawWidth = idrawwidth

Case SCK\_CODE\_NPOLYGON

picDraw.DrawWidth = (sParam1(sData, 6))

picDraw.Line (sParam(sData, 1), sParam(sData, 2))-(sParam(sData, 3),

sParam(sData, 4)), sParam(sData, 5)

If mbServer Then

SendToAllButOriginator SCK\_CODE\_NPOLYGON & sData, viConnection

End If

picDraw.DrawWidth = lblPenSize

Case SCK\_CODE\_UDPOLYGON

picDraw.DrawWidth = (sParam1(sData, 6))

picDraw.Line (sParam(sData, 1), sParam(sData, 2))-(sParam(sData, 3),

sParam(sData, 4)), sParam(sData, 5)



```
Call DrawBCircle(sParam(sData, 1), sParam(sData, 2))
If mbServer Then
    SendToAllButOriginator SCK_CODE_UDPOLYGON & sData,
viConnection
End If
picDraw.DrawWidth = lblPenSize

Case SCK_CODE_PRESENTATION
    lblName.Caption = sckConnect.RemoteHost
Case SCK_CODE_FLIP1
    Call Save
    picFlip.Picture = picDraw.Image
    picDraw.PaintPicture picFlip.Picture, 0, picDraw.ScaleHeight - 1,
picDraw.ScaleWidth, -picDraw.ScaleHeight, , , , vbSrcCopy
    If mbServer Then
        SendToAllButOriginator SCK_CODE_FLIP2, viConnection
    End If
Case SCK_CODE_FLIP2
    Call Save
    picFlip.Picture = picDraw.Image
    picDraw.PaintPicture picFlip.Picture, picDraw.ScaleWidth - 1, 0, -
picDraw.ScaleWidth, picDraw.ScaleHeight, , , , vbSrcCopy
Loop
End Sub
If mbServer Then
    SendToAllButOriginator SCK_CODE_FLIP2, viConnection
End If
Case SCK_CODE_FLIP3
    Call Save
    picFlip.Picture = picDraw.Image
    picDraw.PaintPicture picFlip.Picture, picDraw.ScaleWidth - 1,
picDraw.ScaleHeight - 1, -picDraw.ScaleWidth, -picDraw.ScaleHeight, , , , vbSrcCopy

    If mbServer Then
        SendToAllButOriginator SCK_CODE_FLIP2, viConnection
    End If
Case SCK_CODE_HAMMER
    Call DrawBCircle(sParam(sData, 1), sParam(sData, 2))

    If mbServer Then
        SendToAllButOriginator SCK_CODE_HAMMER & sData, viConnection
    End If
    picDraw.ForeColor = Shape1.FillColor

Case SCK_CODE_HOOK
```





```
Call DrawBCircle(sParam(sData, 1), sParam(sData, 2))
```

```
If mbServer Then
```

```
    SendToAllButOriginator SCK_CODE_HOOK & sData, viConnection
```

```
End If
```

```
picDraw.ForeColor = Shape1.FillColor
```

```
Case SCK_CODE_PRESENTATION
```

```
    lblName.Caption = sckConnect.RemoteHost
```

```
    lblName.Visible = True
```

```
    lblName.Left = sParam(sData, 1) + picDraw.Left
```

```
    lblName.Top = sParam(sData, 2) + picDraw.Top
```

```
    lblName.Visible = False
```

```
If mbServer Then
```

```
    SendToAllButOriginator SCK_CODE_PRESENTATION & sData,
```

```
viConnection
```

```
End If
```

```
Case SCK_CODE_PRESENTATION_STOP
```

```
    lblName.Visible = False
```

```
End Select
```

```
vsString = Mid(vsString, InStr(1, vsString, vbCrLf) + 2, Len(vsString))
```

```
Loop
```

```
End Sub
```



## CHAPTER 5: TESTING

### 5.1 Testing Fundamental

Software errors and failures occur mainly because of inadequate or improper testing. Quality software however demands that software be tested to ensure the system performs according to its specifications and in line with users' requirements and expectations. A number of users are given the opportunity to try the system so as to trace any unforeseen errors or misunderstandings before the system is released. Generally, the goal of carrying out testing for this project is to find faults in the code, seek the root cause and finally fix it throughout the development of the whiteboard system.

This section briefly describes the techniques used to test the program. For clarify our system testing, we simply separate our testing into several step as below.

1. Module/unit testing: Because we have three main module that develop by different members, the first testing is individually and independently on our module and perhaps on smaller units inside our module
2. Integration testing: After the integration of our system, we do another testing on our integration. The integration part is done by Mr. Lee Chee Ching where a Multiple Document Interface (MDI) has been use for this integration
3. Function testing: This path use our most time on testing because lots of error and bug start present especially while connection has been establish. We test all our functions and make sure that they are functioning. We review our requirement specification document to make sure our requirements and expected outcome is fulfill





4. Performance testing: The whiteboard application is a LAN system, where connection must be established. This means that the performance of the system is quite important. On this part of testing, some of the graphic and animation effects of the White Board module have been remove because of unsatisfied performance. As the function testing test for the functioning requirements, we test for the non-functioning requirements of the system on this part
5. Acceptance testing: We test with real data, in the real environment. This carried out to evaluated our final product
6. Installation testing: The installation testing is done through different machine with different version of WINDOWS platform and make sure the deployment is done well

We just simplify our implementation of testing, which descried above to the following simple figure.

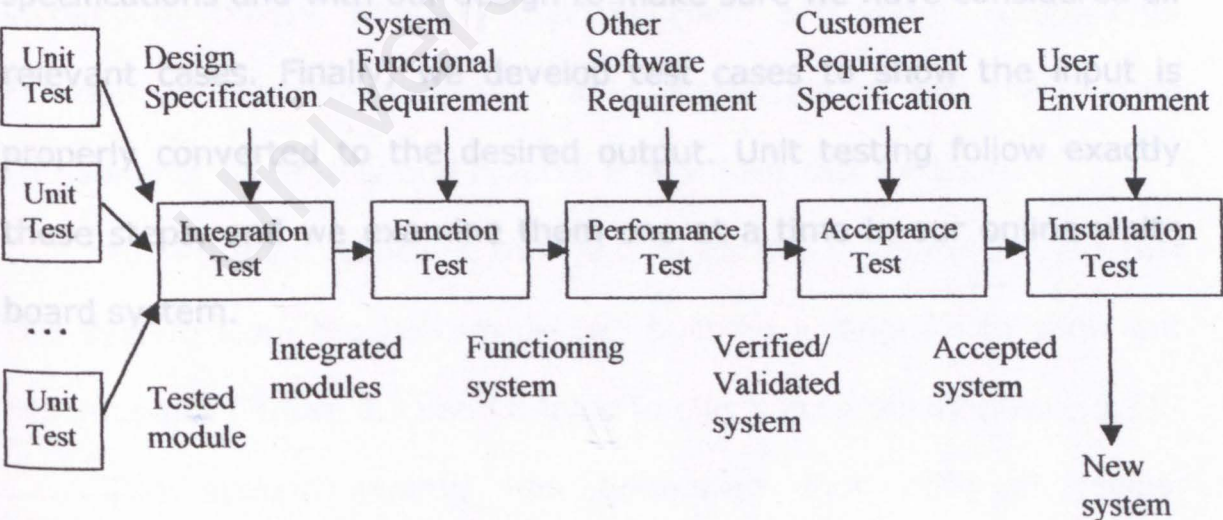


Figure 5.1: Levels of testing



## 5.2 Testing Technique

### 5.2.1 Unit Testing

Unit testing is sometimes called program testing. This type of testing targets verification effort on small unit of codes (subroutines or functions), independent of one another to locate errors. It is the initial testing stage for the completion of each component class. This process enables the tester to detect errors in coding and logical mistake that are contained within that component alone. Testing involving interactions between components are initially avoided and to be carried out later in the bottom-up integration testing.

First, we examine our codes by reading through it, trying to spot algorithm, data, and syntax faults. We may compare the code with the specifications and with our design to make sure we have considered all relevant cases. Finally, we develop test cases to show the input is properly converted to the desired output. Unit testing follow exactly these steps, and we examine them one at a time in our online white board system.





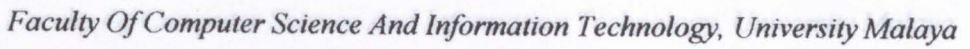
### 5.2.3 Bottom-Up integration

Bottom-Up integration testing is a popular approach of merging components to test the system as a whole. Each component at the lowest level of the system hierarchy is tested individually first. Then the next components to be tested are those call the previously tested ones. This approach is followed repeatedly until all components are included in the testing. Therefore, it is often the most sensible testing for object-oriented program.

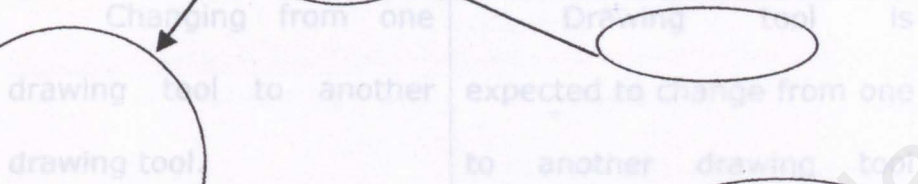
For the case of whiteboard system, objects are combined one at a time with objects or collections of objects that have been tested previously. Messages are sent from one to another and testing ensures that the objects react correctly. It is useful to test those low-level components (classes) in whiteboard system; which often needed to be invoked by others. Knowing that the client and server components function as separate programs, it is reasonable to test the system components separately and reserve a final joint test for them.

For instance, consider the class hierarchy in *figure 3.4*. To test this system from the bottom-up, we built up a diagram to show our testing parts (*Figure 5.2*) and a table to check accordingly (*Table 5.1*).

The system testing has addressed four different levels: functions, classes, clusters (interacting groups of collaborating objects, such as client and server), and the system as a whole. At a minimum,



A test case is a set of input data and expected results that







### 5.2.3 Test Case

A test case is a set of input data and expected results that exercises a system with the purpose of causing failures and detecting faults. Table 5.1 shows the test case.

Action	Expected Result	Fail/Pass
Changing from one drawing tool to another drawing tool.	Drawing tool is expected to change from one to another drawing tool without causing any problems.	
Drawing	Users can their desired drawing. Then, users can draw the drawing on every tools.	
Erasing the drawing on the real time drawing editor.	Users can erase the drawing and other users can see the erase operation.	
Clear all the drawing on the real time drawing editor.	All the drawing on the real time drawing editor will be cleared.	
Saving and printing	Users can their desired save a soft copy or print a	



5.3 Debugging Programs	hard copy	
Setting the draw width and appearance of the drawing.	Users able to set draw width and appearance of their drawing.	
Choosing the desired drawing tool's color.	Users can select a variety of color for their drawing.	
Communicating with other users.	Users can communicate real time with other users.	
Log Off	The user will be disconnected from the whiteboard server.	
Effects functioning as well	User can use the graphic effects every time with no fault and perhaps no delays	
Lock Off by server for presentation	User on server can lock for presentation when a check box is checked. The cursor on every client show for presentation	

Table 5.1: Test Case for White Board module





## 5.3 Debugging Programs

### 5.3.1 Types of bugs

Program can contain three type of bugs, or error:

- Compilation, or syntax, errors – could be an If statement that does not have a corresponding End If statement and could be missing punctuation. Errors are relatively easy to find and fix
- Run-time errors – results from the program trying to do something impossible. Errors are easy to locate.
- Logical error – no syntax error and runtime error but produce incorrect results

### 5.3.2 The VB debugger

The VB debugger allows us to control the program execution in several methods. The following shows the methods that we use in our testing and debugging of our program.

Method	Type of program execution control
Tracing	Runs program line by line, pausing at each until we cause it to execute the next; shows the line-by-line execution of subroutines
Stepping	Runs program line by line, pausing at each until we cause it to execute the next; does not shows the line-by-line execution of subroutines





Breakpoint	Breakpoints are specific places or conditions we set that stop execution, letting us assess the situation at which program execution is paused; for example, if we want to evaluate and change the value of data affecting the program at that point
Immediate and watch windows	Immediate window allows us to view the value of selected variable that help us on testing and debugging

Table 5.2: Method of controlling program execution in the Debugger

PROJECTS DISCUSSION

The whiteboard has some limitations on it. First of all it need a high throughput environment to perform good output. This is because the data flow on this module is very heavy especially on the server side, which a heavy load has been present. Second, the only format for saving the Main Board is in bitmap format.

On development, our team meets a lot of difficulties especially on integration. Before integration, individual program can work smooth because of unheavy loads. After integration, we test our system on 2 computers, 3 computers and mores and meet a significant low throughput on the server path. The workload is increase exponential in adding more computers. We found this on our near ending of development and we choose to not fix it as this is out of our scope in this project. We can fix it with two solutions, as our team has discussed, and will be written here. First, is to change the hole system architecture design from one server implementation to multiple server implementation. This will decrease the workload on the individual, centered server and separate it to others servers. The second solution





is to change the data format on flowing in the socket from sting to byte. Third solution is a very advanced part, which we change our backbone or our principle in using Winsock2 to use the Windows Socket API. This solution is very low level and very difficult in implementation. *Itimedia, paperless, IT world.*

We have removed some of our sub-module that is heavy loading especially the animation and graphic effects on the White Board module. Significant delays have been observed on this types of methods and we choose to remove it to prevent our system in a good throughput. *well at our final testing on it.*

Our team had also compare our developing system with others practical systems and projects. Our system has several significant enhancing especially on the user interface, and information hiding. Our system testers, meanly fellows of ours, are very satisfied on it and we are sure that it is a good review in future to us.

We have chosen to deploy the system with an extra help files attach to it as our users' manual. We using the Html Help Workshop, Html Help Image Editor, and Microsoft Front Page in developing this help file documenting. We have also chosen to package our system onto a CD-ROM when the development ends.

## CONCLUSION

In this project, knowledge of client-server networking and socket programming are combined to develop an online multi-user whiteboard system. The development process began by studying the network requirements of the project and then moved on to implementing the client-server classes necessary for network communication.



Through using this networked whiteboard system, the users are allowed to communicate with people around the world in a fully interactive environment. The users can also treat the system as a stand alone application and use for presentation and teaching purpose in nowadays multimedia, paperless, IT world.

II. As our project end, will not means that our development is also ends. Our team will keep in touch and do some enhancements again on this system. We will try to implement our system in a better throughput as we discuss in the project discussion earlier. The system is functions well at our final testing on it.

Finally, specially thanks to all fellows and friends in helping us on our development especially to our supervisor and modulator.





## REFERENCES

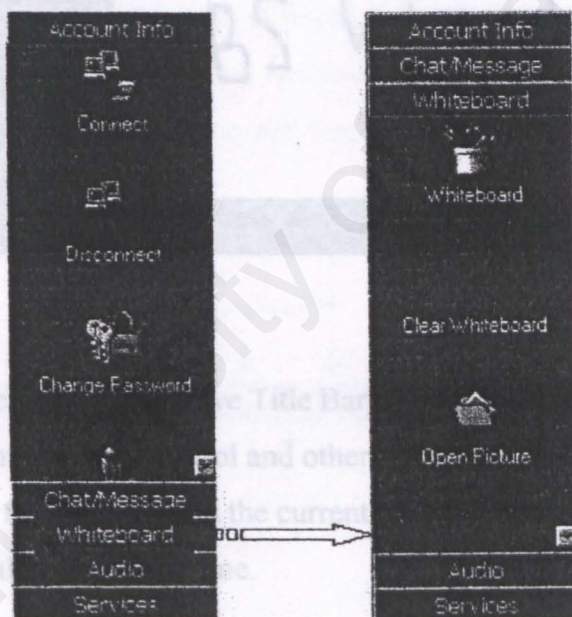
- I. Gary B. Shelly, Thomas J. Cashman, Judy Adamski, Joseph J. Adamski, 1991. *Systems Analysis and Design*. Boyd & Fraser.
- II. Timothy Ramteke, 1999, *Networks*, Prentice Hall publishing
- III. Jeffrey L. Whitten, Lonnie D. Bentley, Kevin C. Dittman, 2001. *System Analysis and Design Methods*, 5<sup>th</sup> ed, McGraw-Hill Irwin.
- IV. Alan R. Apt, 2001. *Software engineering theory and practice*, 2<sup>nd</sup> ed, Prentice Hall International, inc.
- V. Alan M. Davis, 1993. *Software Requirements Objects, Functions, and States*, Prentice Hall, Inc.
- VI.



## APPENDIX

### Getting Start On White Board

On program start, the application will first load in with the default outlook bar which are the Account Info tab. To apply the White Board module, the user must choose to click on the whiteboard button (shown on the figure below). When user click on the white board tab, the windows will drop down two stage to the whiteboard stage. This bar will shown to user whether the connection is already established or idle. Actually the user can threat this white board module as well as a stand alone white board application when the connection is idle.

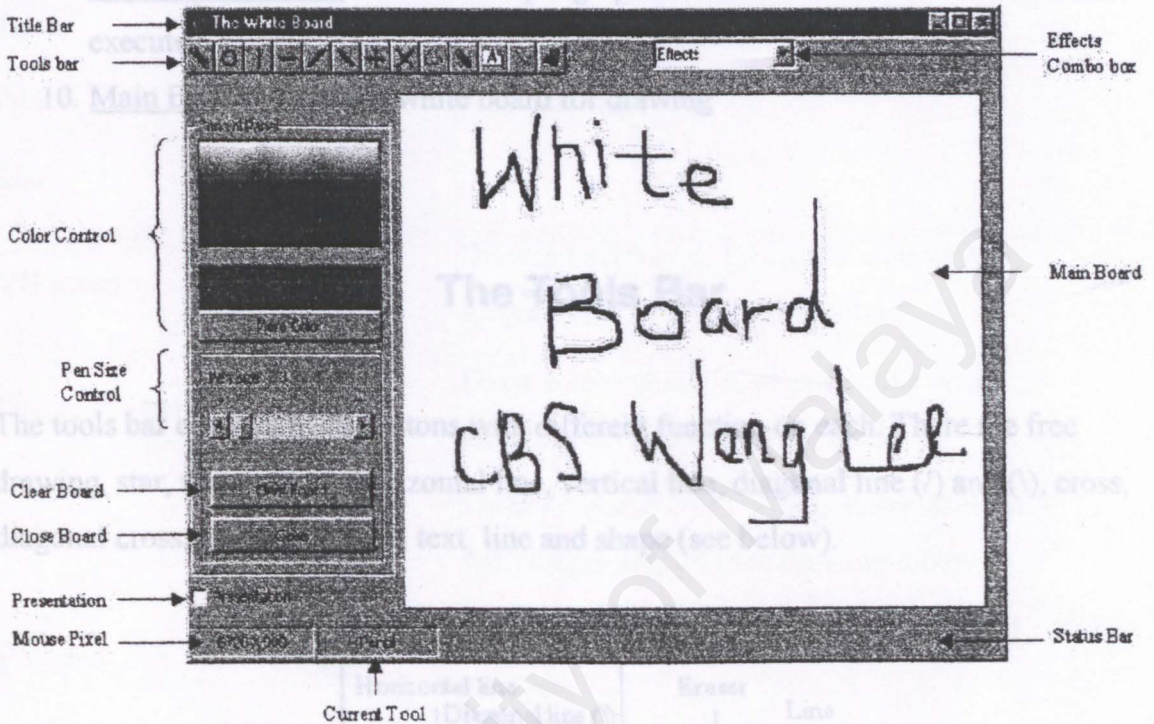


### The Whiteboard

After the user click on the button whiteboard on the outlook bar, the below windows will move in front but with a clear board. The White Board have function akin to a real whiteboard where painting, imaging and pointing are the general functions. we will see



that the WBSYSTEM's whiteboard module will have some added function to a general white board on the later topics. We will discuss several part that include on this module on this page and in detail when we go on deep.



As shown in the figure, the module have Title Bar, Tools Bar, a control frame which consist of pen size control, color control and other control, a presentation checkbox, status bar to show the mouse pixel and the current tool, the Effects combo box, and the Main Board. We will the part one by one.

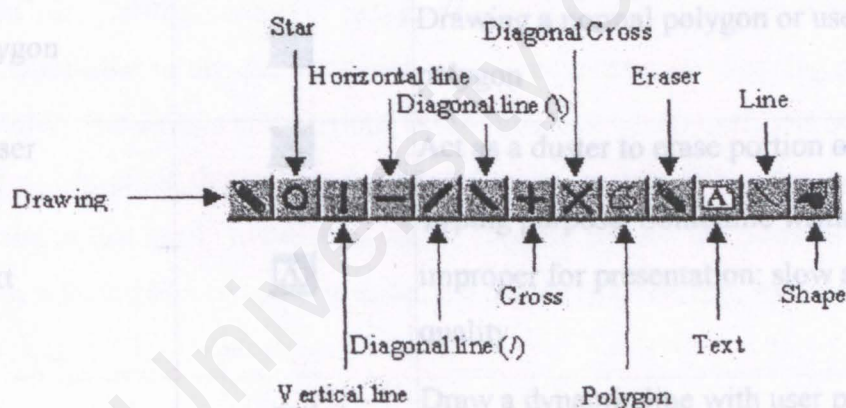
1. Title Bar - shown to user that the name of this module
2. Tools Bar - a multipurpose selectable buttons with function on each
3. Color Control - consist of three parts that is the simple color palette, the selected color for drawing and the advance color palette (the "more color" command button) for user to choose the color for drawing
4. Pen Size Control - Set the pen size on drawing
5. Clear - Clean up the form



6. Close - Close the Module
7. Presentation - call up for presentation ( works only on the connection state)
8. Status Bar - dynamically show the pixel of the mouse when it move over the Main Board and the current using tools (eg. Pencil)
9. Effects Combo box - Several simple graphical effects that the white board can execute
10. Main Board - the main white board for drawing

## The Tools Bar

The tools bar consists of 13 buttons with different function on each. There are free drawing, star, vertical line, horizontal line, vertical line, diagonal line (/) and (\), cross, diagonal cross, polygon, eraser, text, line and shape (see below).

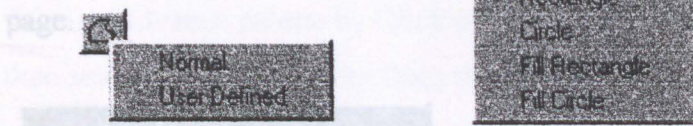












This buttons is simple, but there have two buttons which display a popup menu for selection. There are the polygon and the shape. The polygon will popup a menu for user to choose whether he wants to draw with a normal or user defined polygon, while the shape has 4 selections, that is normal unfilled rectangle, unfilled circle, filled rectangle and filled circle. The function of each button is list down on the follow table.





right click (see figure below). The Effects menu will discuss on the Graphical Effects



Tools	Button (s)	Function
Pencil		Free drawing
Star		
VH Lines		Drawing vertical or horizontal line with length equal to the drawing mode
Diagonal Lines		Draw a / or \ shape of diagonal line with length equal to the drawing mode
Cross		Draw a normal cross '+' or a diagonal cross 'x' with length equal to the drawing mode
Polygon		Drawing a normal polygon or user defined polygon
Eraser		Act as a duster to erase portion of the board
Text		Typing purpose. Sometime writing is improper for presentation: slow and bad quality
Dynamic Lines		Draw a dynamic line with user prefer length and width
Shapes		Drawing shape: rectangle and circle. User can choose to draw it filled or unfill.

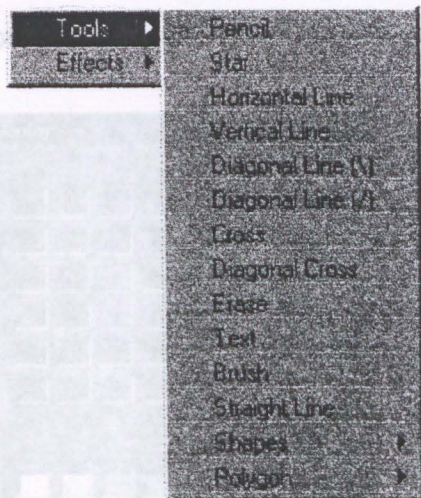
User can also right click on the Main Board to popup a two choice menu: the Tools and Effects. The tools that contain on the Tools Bar is also contain in the Tool menu on





right click (see figure below). The Effects menu will discuss on the Graphical Effects page.

most advance palette by Click on the "define Custom Colors >>" button. Users can then select their prefer color from this advance palette and add their prefer color to



## The Color Control

The Color Control consist of two graphical user friendly interface color palette. Users are not unfamiliar to this palette which usually help users on choosing prefer color easily and usefully. The simple color palette is the little box which users can select their prefer color on just one click. this palette is access directly by user. When the users click on the palette or just easily move their cursor onto the palette, the users will observe that the color selected box (place just under the palette) also change its' color synchronizing with the mouse.



Simple Color Palette



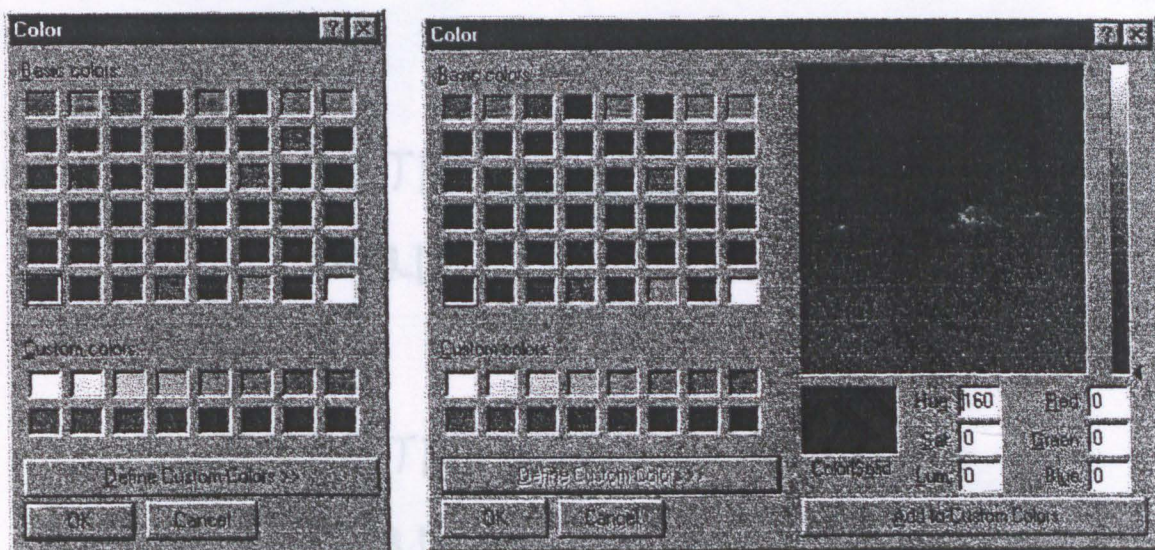
Selected Color



When the users click on the more color button, the advance color palette will popup in front. users can select color from the Basic color palette or the Custom colors palette. If

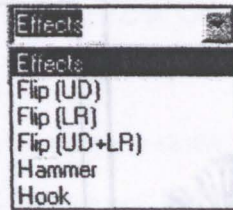


the Basic and Custom palette can not satisfy the user, the user can select the color from the most advance palette by Click on the "define Custom Colors >> " button. Users can then select their prefer color from this advance palette and add their prefer color to custom colors.

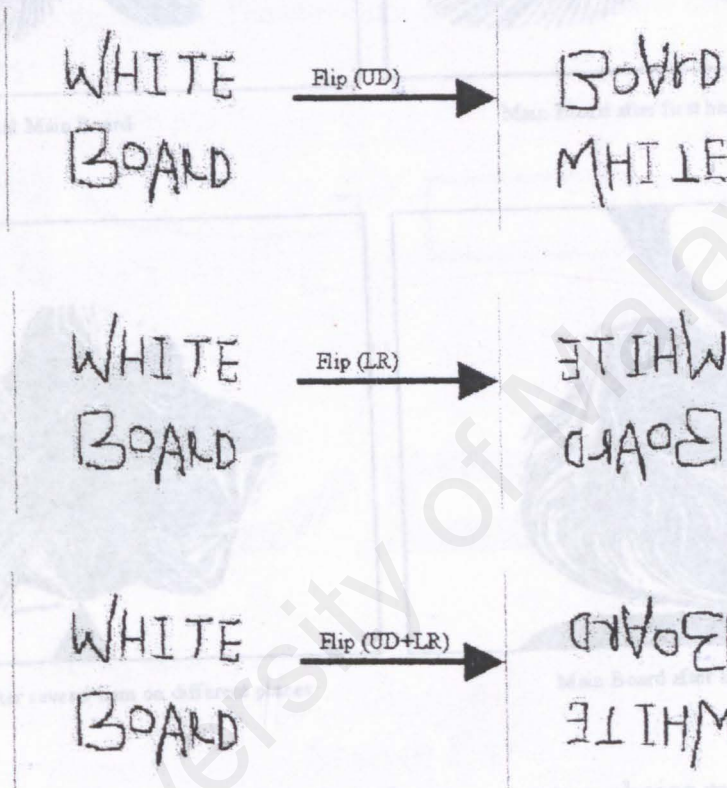


## The Graphic Effects

This version of WBSysstem only contain 5 simple graphical effects (showed below). The Flip (UD) is an effect that flip the Main Board Up-Down, the Flip (LR) flip the Main Board Left-Right and the Flip (UD+LR) Flip the Main Board diagonally. Observation shows that the Flip (UD+LR) is actually Flip (UD) + Flip (LR). The sample of these effects is show on figures below.



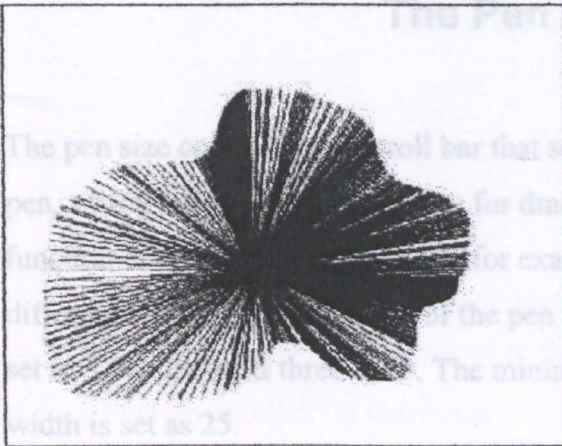
The Effects Combo Box



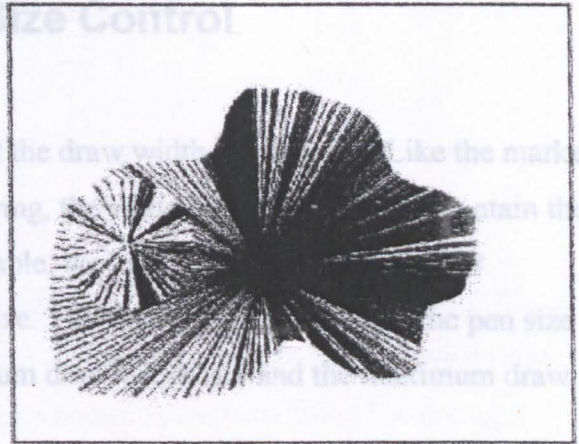
User can also right click on the Main Board to popup a two choice menu: the Tools and Effects. The graphical effects that contain on the Graphical Effects Combo Box is also contain in the Effects menu on right click (see figure below). The Effects menu will

The figure below show the Hammer and the Hook effects. We treat the Main board as a table, when the Hammer take effect, it seems like a hammer ham onto the table (Main Board). Some sample of one ham and several ham is showed below. The last figure is the Hook effect. The Hook is similar to Hammer but only differentiate them is that the Hook can be pull (drag) but the Hammer is not.

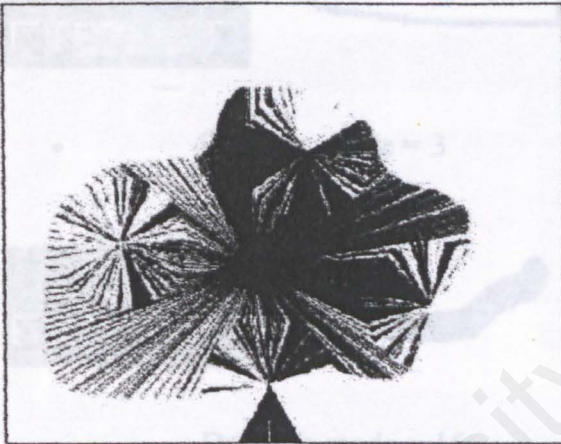




The normal Main Board



Main Board after first ham

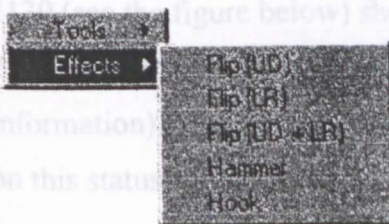


Main Board after several ham on different places



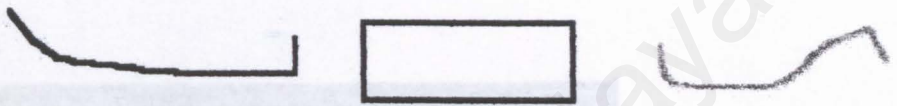
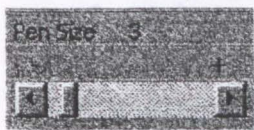
Main Board after Hook

User can also right click on the Main Board to popup a two choice menu: the Tools and Effects. The graphical effects that contain on the Graphical Effects Combo Box is also contain in the Effects menu on right click (see figure below). The Effects menu will discuss on the Tools Bar page.

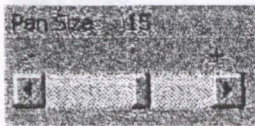


## The Pen Size Control

The pen size control is the scroll bar that set the draw width for drawing. Like the marker pen, which may have different size for drawing, the white board module also contain the function to change the draw width, for example, we look at the figures below for differentiate the drawing width of the pen size. The first three figures show the pen size is set to 3 and the next three is 15. The minimum draw width is 1 and the maximum draw width is set as 25.



- Drawing mode = 3



- Drawing mode = 15

### Status Bar

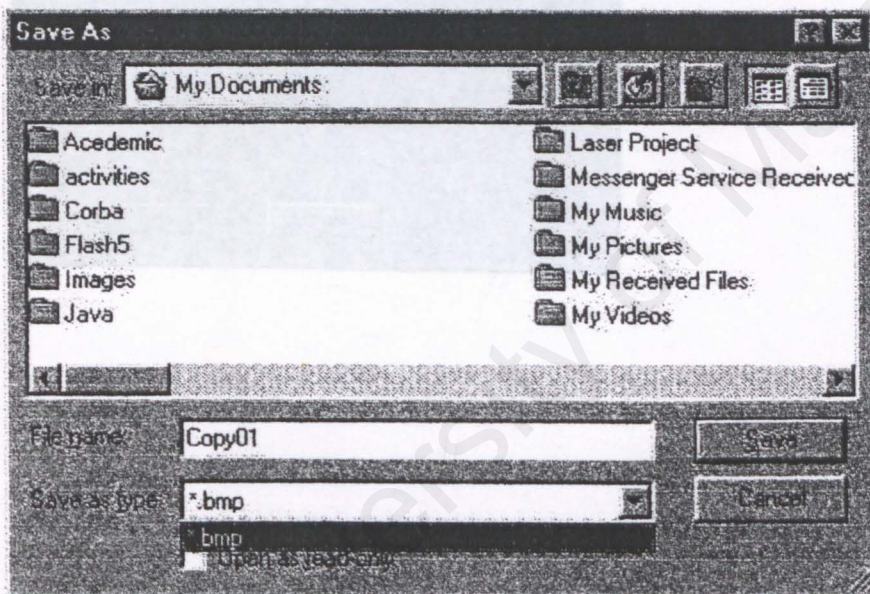
The status bar is the bar at the module that show the current status of program. The status bar consist of two section: the mouse pixel and the current tool. The mouse pixel will show the current mouse pointer (if the cursor is on the Main Board), for example, 285 X 2130 (see the figure below) shown that the x-pixel of the mouse is 285 and the y-pixel 2130. Pencil is the current tool. There have lot of other tools (see [tools bar](#) for more information). When users choose a tool, the tool's name will immediately show up to user on this status bar as below.



## Saving and Printing

After users is drawing on the Main Board, users can choose to make a save copy or print a hard copy of this board for future review.

For saving, the only format for user to make their save copy is the 16 bit "\*.bmp" file for this version. As we view from the WhiteBoard outlook bar, we can find a Save button there. When users click on that button, a popup common control dialog for saving is popup which is not unfamiliar to us. We can just choose the location and give the file name for saving.



For Printing, Users click on the Print button. A simple Print the picture window is popup for users to setup a simple properties for the printer to make a print copy. Users click on the Print button and the program will contact the allocated printer to make the copy, for e.g. , if the default printer is set to Canon BJC-265 SP on the LPT1, the following message box will popup.

